Multinomial language learning

Raaijmakers, S.A.

Publication date:
2009

Document Version
Publisher's PDF, also known as Version of record

Multinomial Language Learning

Investigations into the Geometry of Language

Stephan Raaijmakers

Multinomial Language Learning

Investigations into the Geometry of Language

Proefschrift

ter verkrijging van de graad van doctor aan de
Universiteit van Tilburg,
op gezag van de rector magnificus,
prof. dr. Ph. Eijlander,
in het openbaar te verdedigen
ten overstaan van een door het college voor
promoties aangewezen commissie
in de aula van de Universiteit
op dinsdag 1 december 2009 om 14.15 uur

door

Stephan Alexander Raaijmakers

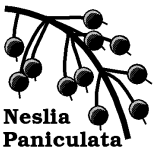
geboren op 29 maart 1964 te Hazerswoude

Promotores:

Prof. dr. A.P.J. van den Bosch
Prof. dr. W.M.P. Daelemans

Beoordelingscommissie:

Dr. V. Hoste
Prof. dr. F.M.G. de Jong
Prof. dr. E.O. Postma



Taaluitgeverij Neslia Paniculata
Uitgeverij voor Lezers en Schrijvers van Talige Boeken
Nieuwe Schoolweg 28, 7514 CG Enschede, The Netherlands

SIKS Dissertation Series No. 2009-40
The research reported in this thesis has been carried out under the auspices of SIKS, the Dutch Research School for Information and Knowledge Systems.

TiCC Dissertation Series No. 08

This work has been partially supported by the European IST Programme Project FP6-0033812.

Cover wittily inspired by Chris Manning and Hinrich Schütze, *Foundations of Statistical Natural Language Processing*, MIT Press, 1999.

Printed and bound by PrintPartners Ipskamp, Enschede.

ISBN 90-75296-15-0

Contents

Acknowledgments

This thesis is the final outcome of a long and complex computation, involving a multitude of operating systems, a few generations of hardware, a couple of forking paths, and quite some backtracking.

Many people have contributed more than they know to this work. First of all, I am greatly indebted to my supervisors Antal van den Bosch and Walter Daelemans for encouraging me, and wading without any complaint through myriads of ideas, papers, thesis proposals, and subversive implementations of memory-based classifiers. Their patience, advice and support has been of utmost importance for the completion of this thesis.

As a student at Leiden University, my teachers Teun Hoekstra, Harry van der Hulst, Jan Kooij, and Vincent van Heuven introduced me to the formal concepts of linguistics with a rigor that one would normally expect in the natural sciences. Their empirical attitude and curiosity have deeply influenced me. My master thesis supervisor Joan Baart was crucial in my late stage development as a student. His careful analytical approach to complex computational issues still serves as a style guide for me. My work as a student in Saarbrücken on Symantec's Q&A –proudly proclaimed as the world's first natural language database interface– was my first realistic encounter with the industrial application of computational linguistics. I enjoyed every minute of it, thanks to the hospitality of Erwin Stegentritt and Axel Biewer. The cooperation initiated with Michael Moortgat during the final stages of my studies has greatly influenced my thinking about language and logic.

Following my time in Leiden as a student, I virtually entered post-graduate education by the inspiring environment of the Institute for Language Technology and Artificial Intelligence (ITK) in Tilburg, where casual talks over a cup of coffee could easily lead to earth-shattering new theories of language and computation. People like Erik Aarts, René Ahn, Harry Bunt, Jan Jaspars, Hap Kolb, Reinhard Muskens, Elias Thijsse, Leon Verschuur, and many others made this

place the national hotspot for logic, language and discourse analysis. Contacts and collaboration with people from the ILLC group of the University of Amsterdam (especially Johan van Benthem, Herman Hendriks, and Dirk Roorda) were very stimulating.

During my subsequent stay at the Institute for Dutch Lexicology in Leiden, I really got my hands dirty on corpus annotation, European lexicology projects and Hidden Markov part-of-speech taggers. Somewhat in the spirit of a linguistic institute, I added Perl, Java, C(++), and a touch of Python to my programming languages repertoire. The energy and enthusiasm of Truus Kruyt and my colleagues of the Taalbank has always been a great stimulus.

At TNO, Wessel Kraaij has been of substantial moral support. His friendship, expertise and optimism were vital factors for the completion of this thesis. Anita Cremers, Franciska de Jong, David van Leeuwen, Ruud van Munster, Wilfried Post, Jeroen van Rest, Nellie Schipper, Jan Telman, Andy Thean, Khiet Truong, my colleagues from the Multimedia Technology department and many others have made TNO a splendid place to work. I thank my former business unit manager Marcel Teunissen, our director for knowledge Erik Huizer and my manager Dick van Smirren for their support and interest, and for making things possible. The cooperation of TNO with the Universities of Amsterdam, Delft and Edinburgh in both national and European projects has led to fruitful collaboration with Jan van Gemert, Cees Snoek, Marcel Worring, Alan Hanjalic, Inald Lagendijk, Martha Larson and Theresa Wilson.

Finally, the support of my good friends and family has been invaluable.

I dedicate this thesis to my daughters Jasmijn and Sanne. May the distance between us, either Euclidean or geodesic, always be minimal.

Amsterdam, 2009 *Stephan Raaijmakers*

Part I

Geodesic Models of Document Geometry

Chapter 1

Introduction

Nowadays, with the enormous abundance of electronically available texts, manual analysis of documents is no longer feasible. Yet, document analysis is necessary for access to documents beyond simple keyword search. For instance, the classification of a topic of a document greatly facilitates the retrieval of related documents. The weblog monitor website Technorati[1] reported as per June 2008 a massive amount of 133 million weblogs across the world, with 900,000 blog postings per 24 hours. This current orientation of the Web towards self-publishing, with immensely popular social media such as Twitter, MySpace, Facebook and Flickr, clearly illustrates the urgency for automated procedures that accurately monitor, analyze and label the information in online repositories. Due to the large number of documents involved, precision becomes increasingly important: generally speaking, any system facilitating a user searching a large set of documents should minimize the number of retrieved documents on the basis of the user's query to the smallest possible set of relevant documents. Therefore, precise document analysis is crucial.

The automated analysis of text (or more broadly, linguistic content) has a long tradition, and has in its early days benefited from the exploratory work of e.g. Harris [1959] and Hillel [1964], in an era (the fifties and sixties of the previous century) that identified in particular automated (machine) translation as a desideratum. The field of computational linguistics, while originally drawing heavily upon linguistics and statistics, has demonstrated the last two decades an increasing orientation towards *machine learning*: the automated analysis of data using learning algorithms, trained on sample, usually hand-performed analyses. A trained learning algorithm (or *classifier*) performs a *classification* of an object (such as a document) into one or more *classes*, by analogy to the data it was trained on. This classification can subsequently be assigned as *meta-data* to the original datum. The commonly held opinion is nowadays that linguistic analysis can be seen as a classification process, and that complex analyses can

[1] http://www.technorati.com

be built up from less complex subordinate analyses (see e.g. Daelemans and van den Bosch [2005]).

Classification is founded upon a notion of distance: test data is compared to exemplary training data using certain distance metrics. The examples that most closely resemble the test data determine its classification. This thesis addresses the automated analysis of text documents from a machine learning perspective, and specifically focuses on the issue of distance metrics. Central in this work is a set of recently proposed machine learning algorithms that exploit the intrinsic geometry of the data space text documents are situated in, given a certain statistical representation of these documents. This data space has been shown to possess *geodesic* structure: it is a curved space much like Earth, and distance metrics between documents should take into account curvature for accurate measurements. We will formally analyze this type of algorithms, propose extensions, and involve them into a number of new applications. In particular, we will closely monitor the underlying distance metrics, and demonstrate under which circumstances they reach optimal performance. The research topics of this thesis are outlined in detail in Section 1.1.

1.1 Research questions

The problem of establishing similarity between documents, which is the central problem underlying document retrieval and document classification, is traditionally solved by measuring distances in the vector space constituted by the vector representations of a set of documents. The determination of the topic of an unlabeled test document is a case in point: given e.g. a 1-nearest neighbor classifier operating in vector space, we simply look for the labeled training document of which the feature vector has minimal Euclidean distance to the feature vector of the test document. The test document obtains the class label of the corresponding training document.

The classic view on vector space is that it is *Euclidean*: a flat world without curvature, where the distances between objects (document representations) are to be measured along straight lines. Recently, this Euclidean assumption has been challenged. The work of John Lafferty and Guy Lebanon (e.g. Lafferty and Lebanon [2005]) has triggered an active research field concerned with non-Euclidean geometry of document space. Their work demonstrates that a very simple document representation, the so-called *L1-normalization of word frequencies*, yields an embedding of documents in a curved information space, known as a *Riemannian manifold*. This particular manifold corresponds to the parameter space of the multinomial distribution. On Riemannian manifolds in general, distances should not be measured with straight lines, but, just like between points on Earth, along curves with geodesic distance measures.

We will evaluate this geodesic approach to document space geometry, which

we shall dub 'multinomial language learning'. After establishing the performance of geodesic classifiers on a number of document classification tasks, we address the following problem:

> **Research question 1 (heterogeneous information):** *Multinomial language learning is based on statistical representations (L1-normalized frequencies) of strings, such as words, or word n-grams. An implicit assumption behind the approach is that these strings form a homogeneous distribution, consisting of e.g. all separate words, or all word bigrams, but not both. Can we mix heterogeneous families of strings under this approach, such as word unigrams and bigrams?*

Multinomial language learning, while suitable for documents, does not lend itself easily to limited context tasks, or sequentially organized, feature-based tasks, like part-of-speech tagging: these tasks usually consist of small, fixed-size windows over ordered feature sequences, where the notion 'frequency' (the crucial ingredient of L1-normalization) does not come into play. Therefore, we address

> **Research question 2 (sequential and limited context tasks):** *How can we apply multinomial language learning to tasks with a limited amount of context, or even sequentially organized tasks?*

The Euclidean point of view ignores the intrinsic geometry of objects, and just measures angles between vectors in a flat hyperspace. The multinomial language learning view on document geometry offers an alternative: it assumes that Euclidean distance only comes into play on a small scale, between neighboring points where the effect of curvature is negligible. The question we would like to raise is:

> **Research question 3 (high entropy data and geodesic distance):** *What is the effect of the distance between data points on the performance of geodesic distance measures?*

Specifically, we will study degenerate cases where the geodesic distance measure is no longer accurate, and, in fact, collapses into the Euclidean distance measure. The triggers a corollary question:

> **Research question 4 (hybrid geometry):** *Is document space under the L1-based representation schema best modeled using curved manifolds only, or are hybrid geometrical (Euclidean and geodesic) representations desirable?*

We will investigate the possibility of hybrid geometry for document space, and the conditions of optimal combination of Euclidean and geodesic information:

Research question 5 (classifier calibration): *How can classifiers be calibrated in order to yield optimal performance when applied to document spaces with hybrid geometry?*

The problem statement of this thesis therefore is three-fold:

1. Can we extend the standard multinomial language learning apparatus to heterogeneous data, and sequential, limited context classifications tasks? (Research questions 1 and 2)

2. Can we motivate the existence of hybrid geometry in L1-normalized document representations? (Research questions 3 and 4)

3. If such hybrid geometry can indeed be motivated, how can we calibrate classifiers operating on this document space such that their performance is optimized? (Research question 5)

The answers to research question 3, 4 and 5 serve to support our central theorem:

Depending on the entropy of local neighborhoods, the space of L1-normalized document representations possesses both Euclidean and geodesic structure, and classifiers should be aware of this hybrid structure for optimal performance.

1.2 Thesis outline

The thesis consists of two major parts. In part I, *Geodesic Models of Document Geometry*, we are concerned with the application of geodesic kernels to a number of document classification tasks, predominantly from the sentiment mining field (detection and classification of sentiments in text). In Chapter 2, we present a bird's eye overview of machine learning, introducing the important concepts and algorithms for our work. This introduction contains a novel algorithm for the estimation of hyperparameters of machine learning algorithms. We demonstrate in Chapter 3 the adequacy of geodesic kernels for the classification of sentiment at the sentence level (subjectivity classification) as well as at the document level (polarity classification). Subsequently, we propose two types of extensions. In Chapter 4, addressing research question 1, we propose an interpolation strategy for combining multiple sources of information. We formally show that this interpolation strategy corresponds to a stratagem known as *manifold regularization*, and a so-called *pullback* operation on a manifold. In Chapter 5, we address research question 2 and extend the multinomial approach to sequential tasks without explicit document structure. We formally show that this extension also corresponds to a pullback operation.

Part II of the thesis, *A Back-Off Model for Document Geometry*, is concerned with finding evidence for the hypothesis that document space has, under certain conditions, *hybrid* geometry: both geodesic (explicitly curved) and Euclidean (flat). We start out in Chapter 6 with a formal proof that under the unfavorable condition of maximum entropy data, a geodesic distance measure known as the *negative geodesic kernel* becomes isometric to the Euclidean kernel. This chapter investigates the relationship between data entropy and distance, and thus answers research question 3.

In Chapter 7, we empirically confirm the presence of both Euclidean and geodesic structure in L1-normalized data, on the basis of evidence from feature selection and manifold denoising experiments (research question 4). We develop a *dimensionality reduction* (feature selection) method based on heterogeneous Laplacian Eigenmaps. We demonstrate for a number of data sets that standard Euclidean Laplacian Eigenmaps, when combined with geodesic Laplacian Eigenmaps, lead to better results than single-geometry variants. In a similar spirit, we propose a manifold denoising method that removes both Euclidean and geodesic noise from Riemannian manifolds. The combined approach yields optimal performance in most cases. Finally, in Chapter 8, we answer research question 5 and develop a method for classifier calibration. This method can be used to identify hard cases that cannot be classified automatically with a prespecified accuracy. We subsequently generalize this method, by showing that it can be used to identify intrinsically Euclidean and geodesic parts of data on the basis of the entropy of local neighborhoods of test points. This elaborates on the formal result of Chapter 6, by estimating thresholds that restrict the application of the negative geodesic distance to low or moderate entropy data, and leads to an operationalization of the hybrid view on document space geometry: it implements a decision procedure for switching between two alternative representations of the same data. We demonstrate that the usefulness of the calibration technique is a consequence of the established relation between entropy and distance between datapoints in a manifold.

Appendix A contains an explanation of the formal concepts relevant to the study of manifolds and information geometry. Appendix B describes the formal apparatus necessary to evaluate classifiers. We have deferred the description of all algorithms to Appendix C in order to improve readability. In Appendix D, as an aside, we outline a connection of our work with the rapidly emerging fields of quantum information science and quantum machine learning.

1.3 Research methodology

This is an evidence-based thesis, in which we will attempt to gather answers for our research questions based on empirical evidence. Consistently, we will use hand-annotated (i.e. pre-classified) datasets, and split them up into separate

training, test and development partitions. The development partitions are used to tweak eventual parameters of the machine learning methods to be applied, using specialized search algorithms (see Section 2.2). These partitions are usually split up in one or more training/test parts themselves. Once parameters have been fixed, the actual application of the machine learning algorithms to the remaining training and test partitions takes place. Results are reported using well-known measures from the fields of information retrieval and machine learning, as laid out in Appendix B. Whenever two or more learning algorithms are compared, statistical significance and correlation tests are carried out in order to measure performance differences.

Chapter 2

Machine learning: algorithms and data representation

In this chapter, we introduce the major concepts of machine learning, and discuss the formalism we investigate in this thesis: multinomial classifiers using geodesic distance measures. We start with a general overview of machine learning in Section 2.1, with emphasis on the family of classifiers known as *Support Vector Machines*. In Section 2.2, we will present and evaluate a novel algorithm for the tuning of classifiers. We will make use of this algorithm in a number of experiments discussed in Chapter 4. Section 2.3 discusses the ubiquitous vector space model for document classification, a representational space that is intimately linked to Euclidean distance measures. In Section 2.4 we introduce the general concept of the multinomial manifold, the geometric backbone of multinomial classifiers. In Section 2.5 we discuss two non-Euclidean, geodesic kernels operating on the multinomial manifold: the Information Diffusion Kernel, and the Negative Geodesic Distance kernel.

2.1 Machine learning

Machine learning is an active field of research, and is roughly interdisciplinary between statistics, information science, cognitive psychology, and mathematics. The main research question addressed by machine learning is to find both accurate and parsimonious models of learning, which can be used to teach the discrimination of different objects to a machine. Well-known examples are face and speaker recognition (recognizing persons on the basis of visual and acoustic cues), image classification (labeling images with descriptive keywords), intrusion detection (classification of network behavior as deviant or normal), and a wide variety of linguistic applications such as document classification (assigning topics to documents), part-of-speech tagging (assigning parts of speech to words in a text), syntactic parsing (assigning phrasal structure to sequences of words), and sentiment classification (labeling 'emotions' or opinions in e.g. blog posts).

Formally, a model of learning is a complex function that maps descriptions of objects (*instances*) I to a set of classes C:

$$f : I \mapsto C \tag{2.1}$$

The instance space I is a *feature space*: a vector space of descriptors (features) that describe a certain aspect of an object using a well-defined vocabulary. Every $i \in I$ is a vector of feature values. Binary classification problems limit C to $\{-1, +1\}$. One-class classification problems also exist: sometimes, a complementary second class is too rare to provide sufficient training data (e.g. data representing the malfunctioning of an almost error-free system) or the complementary class is too heterogeneous (e.g. certain content that is *not* of interest to a certain user). One-class classifiers are solely trained on data representing the single observable class, which is modeled as a coherent group (Tax [2001]). In machine learning, classes usually consist of discrete symbols (class labels), but can be real-valued as well, in which case we speak of *regression*.

If the function f is derived from a set of pre-labeled (and usually hand-checked) examples, we call any method to derive f from the examples *supervised*. The set of input examples to derive f is called the *training set*. Alternatively, f can be deduced from unlabeled data as well, in which case f is called *unsupervised*, or from mixtures of labeled and unlabeled data, in which case we call f *semi-supervised*. Transductive learners (e.g. Joachims [1999]) are test set-specific classifiers that use unlabeled information from a specific test set to better approximate the class boundaries dictated by the labeling in the training data.

Usually, f is parametrized, needing adjustment of a certain set of controls called *hyperparameters*. The process of finding values for these hyperparameters, which directly control the learning process, is called *hyperparameter estimation*. We return to the problem of hyperparameter estimation in Section 2.2, and propose a novel approach to the problem based on a stochastic sampling method.

In addition to hyperparameter estimation, feature selection can be performed to eliminate noise from training and test data. Both hyperparameter estimation and feature selection are part of the training process. A fully *trained* version of f, then, is called a *classifier*. It can be applied to a set of *test data*, and will produce assignment of each of the test examples to one or more classes, using the knowledge it inferred from the training data. Typically, f is constrained during the learning process by a *loss function* measuring the discrepancy between the correct class of a datapoint x and the prediction of f for x.

After training, the classifier f will have access to a set of instantiated variables adjusted during the training process. These variables (e.g. weights for features, or instances) are called *parameters*. Together, sometimes coupled with a selection of important examples from the training data, they make up the *model* for f. Models can be derived as the product of training, in which case

they are a succinct, statistical approximation of the set of training examples. A model is a *hypothesis*: it is an estimate of a *target function* represented partially by the labeled training data. Finding the best hypothesis is the solution of the learning problem. Model-constructing classifiers are called *eager* learners. So-called *lazy* learners do not infer a model from their training data, but, instead, construct separate models on the fly for test points during classification. For these learners, learning basically is storage without abstraction. For this reason, these methods are also called *memory-based* (Daelemans and van den Bosch [2005]).

Yet, even if it is possible to accurately model the training data with a good hypothesis, this is no guarantee that this hypothesis can be fruitfully applied as an accurate classifier to new, unseen test data. So, in addition to being simple and accurate, we want our hypothesis to be able to *generalize* to new data outside the training data. Hypothesis selection methods balancing complexity and accuracy should be used here. For instance, *rote classifiers*, classifiers that have zero test error on their training data, but perform erroneously on test data, can be easily constructed by optimally complex models storing all training data. These models are *overfitting*, by becoming too specifically tuned to the training data used. Ockham's razor-based selection or *model pruning* principles (such as Minimum Description Length (MDL; Rissanen [1983]) can be applied to select the simplest yet most accurate hypothesis from the hypothesis space. Here, the notion of *class separation* comes into play. A classifier learning to discriminate between objects of, say, classes y_1 and y_2, will need to find an optimal boundary b between y_1 and y_2 such that b maximizes the distance between objects of class y_1 and y_2 everywhere in the model representation of the training data. This type of boundary will have minimal overfitting, and is known as a *large margin*[1]. An example of large-margin based classifiers are Support Vector Machines (Cristianini and Shawe-Taylor [2000]). Building upon the initial approach of Rosenblatt's Perceptron from 1957 (Rosenblatt [1958]), Support Vector Machines attempt to find a linear function performing a large margin separation of input data. In Figure 2.1, two classes are separated by three boundaries called *hyperplanes*: linear functions that are described by $\mathbf{w} \cdot \mathbf{x} + b$, with $\mathbf{w}$ a weight vector, $\mathbf{x}$ an n-dimensional vector in $\mathbb{R}^n$, and $b \in \mathbb{R}$. The hyperplanes $\mathbf{w} \cdot \mathbf{x} + b = 1$ and $\mathbf{w} \cdot \mathbf{x} + b = -1$ are maximally apart. The intermediate hyperplane $\mathbf{w} \cdot \mathbf{x} + b = 0$ is the maximum hyperplane or margin: it has maximal distance to any of the datapoints in the two classes.

Let the Euclidean distance between two datapoints (vectors in a n-dimensional space) be defined as

$$\| \mathbf{x} - \mathbf{y} \| = \sqrt{\sum_{i}^{n} (\mathbf{x}_i - \mathbf{y}_i)^2} \tag{2.2}$$

[1]The principle of large margin separation is not universally the best option; sometimes, class boundaries are quite sharp, and do not demand large margin separation.

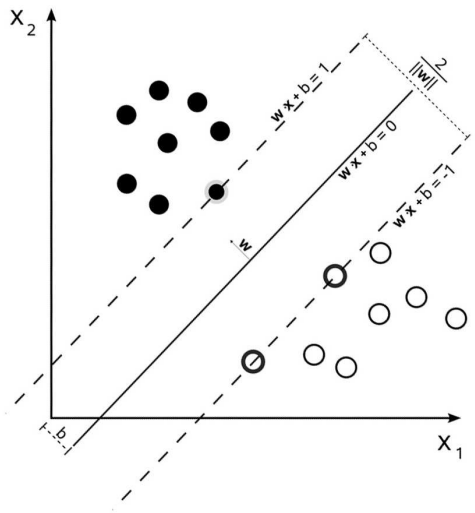


Figure 2.1: Large-margin binary classification.

The Euclidean norm (or length) or a vector is defined as

$$\| \mathbf{x} \| = \sqrt{\sum_{i}^{n} \mathbf{x}_i^2} \tag{2.3}$$

It can be shown that the Euclidean distance between the two border hyperplanes is $\frac{2}{\|\mathbf{w}\|}$, so minimizing $\| \mathbf{w} \|$ increases distance. The proof of this is straightforward.

Let $\mathbf{x}$ be any point on the hyperplane $\mathbf{w} \cdot x + b = 1$, and $\mathbf{y}$ any point on $\mathbf{w} \cdot x + b = -1$. Proving that the weighted difference between $\mathbf{x}$ and $\mathbf{y}$, $\mathbf{w}(\mathbf{x} - \mathbf{y})$ equals 2 entails that the distance between $\mathbf{x}$ and $\mathbf{y}$, $\| \mathbf{x} - \mathbf{y} \|$, (and hence the distance between the two hyperplanes they lie on) is $\frac{2}{\|\mathbf{w}\|}$.

2.1.1. PROPOSITION. $\mathbf{w}(\mathbf{x} - \mathbf{y}) = 2$ *implies* $\mathbf{x} - \mathbf{y} = \frac{2}{\mathbf{w}}$.

Proof We know: $\mathbf{w} \cdot \mathbf{x} + b = 1$ and $w^T \cdot \mathbf{y} + b = -1$. Then we have $\mathbf{x} = \frac{1-b}{\mathbf{w}}$ and $\mathbf{y} = \frac{-1-b}{\mathbf{w}}$. Starting from $\mathbf{w}(\mathbf{x} - \mathbf{y}) = \mathbf{w} \cdot \mathbf{x} - \mathbf{w} \cdot \mathbf{y}$, since $\mathbf{w}(\frac{1-b}{\mathbf{w}}) = 1 - b$ and $\mathbf{w}(\frac{-1-b}{\mathbf{w}}) = -1 - b$, we obtain $1 - b - (-1 - b) = 2$. From this, it follows that $\mathbf{x} - \mathbf{y} = \frac{2}{\mathbf{w}}$, hence $\| \mathbf{x} - \mathbf{y} \| = \frac{2}{\|\mathbf{w}\|}$ as desired. ■

Every datapoint $\mathbf{x}_i$ in one of the two classes $\{+1, -1\}$, written as C_+, C_-, is constrained by

$$\begin{array}{l} \mathbf{w} \cdot \mathbf{x}_i + b \geq 1, \forall \mathbf{x}_i \in C_+ \\ \mathbf{w} \cdot \mathbf{x}_i + b \leq 1, \forall \mathbf{x}_i \in C_- \end{array} \tag{2.4}$$

which leads to the following optimization problem for Support Vector Machines:

$$\begin{array}{l} \forall 1 \leq i \leq n, \textit{find optimal } \mathbf{w}, \mathbf{x} \textit{ that minimize } \parallel \mathbf{w} \parallel \textit{ under the constraints} \\ \mathbf{w} \cdot \mathbf{x}_i - b \geq 1, \forall \mathbf{x}_i \in C_+ \textit{ and } \mathbf{w} \cdot \mathbf{x}_i - b \leq -1, \forall \mathbf{x}_i \in C_- \end{array} \tag{2.5}$$

This has an equivalent formulation (c_i the class of point $\mathbf{x}_i$ in the training data):

$$\begin{array}{l} \forall 1 \leq i \leq n, \textit{find optimal } \mathbf{w}, \mathbf{x} \textit{ that minimize } \parallel \mathbf{w} \parallel \textit{ under the constraint} \\ c_i(\mathbf{w} \cdot \mathbf{x}_i - b) \geq 1 \end{array} \tag{2.6}$$

which allows for a *dual formulation*: given

$$\begin{array}{l} \mathbf{w} = \sum_i \alpha_i c_i \mathbf{x_i} \\ \sum_i \alpha_i c_i = 0 \\ \alpha_i \geq 0 \end{array} \tag{2.7}$$

find

$$\max \sum_{i=1}^{n} \alpha_i - \frac{1}{2} \sum_{i,j} c_i c_j \alpha_i \alpha_j \mathbf{x}_i^T \mathbf{x_j} \tag{2.8}$$

This is a *Lagrangian dual problem* (Cristianini and Shawe-Taylor [2000]). Every optimization problem

$$\begin{array}{lll} \text{Find the minimum of } f(\mathbf{w}) & (\mathbf{w} \in \mathbb{R}^n) & \\ \text{under constraints} & g_i(\mathbf{w}) \leq 0, i = 1, \ldots, k & \\ & h_i(\mathbf{w}) = 0, i = 1, \ldots, m & \end{array} \tag{2.9}$$

has a Lagrangian dual formulation

$$\begin{array}{ll} \text{maximize} & \theta(\alpha, \beta) \\ \text{under constraint} & \alpha \geq 0 \end{array} \tag{2.10}$$

where

$$\begin{array}{ll} \theta(\alpha, \beta) & = \inf_{\mathbf{w} \in \mathbb{R}^n} L(\mathbf{w}, \alpha, \beta) \\ & = f(\mathbf{w}) + \sum_{i=1}^{k} \alpha_i g_i(\mathbf{w}) + \sum_{i=1}^{m} \beta_i h_i(\mathbf{w}) \end{array} \tag{2.11}$$

Only for the datapoints $\mathbf{x}_i$ that lie closest to the margin, the α weights are non-zero. The other datapoints receive zero weights as they do not play a role in computation of the optimal hyperplane. The datapoints with non-zero weights are called the *support vectors*.

The optimal hyperplane can be represented parsimoniously by the data points that lie on its margins. Notice the use of dot products in the dual formulation. Swapping these dot products with non-linear functions leads to implicit expansion of the input space (the original features space) to a high dimensional feature space (the space that emerges after the implicit expansion). For instance, the feature space derived from the input space $\{x, y\}$ for a polynomial function $f(x, y) = (x + y)^2$ would be $\{x^2, 2xy, y^2\}$. Data that is linearly inseparable in the low dimensional input space hopefully becomes linearly separable in feature space. The implicit expansion of low dimensional input space to high dimensional feature space is known as the *kernel trick*. The expansion is never carried out explicitly, but occurs as a side-effect of performing the non-linear computations. Both the standard dot product and its non-linear variants are called *kernels*. A kernel is a similarity function that computes a similarity score for two input feature vectors. The usual operations involved in these computations originate from matrix algebra, such as the dot product:

$$\mathbf{x} \cdot \mathbf{y} = \sum_{i=1}^{n} \mathbf{x}_i \mathbf{y}_i + \ldots + \mathbf{x_n} \mathbf{y}_n \tag{2.12}$$

which can be equivalently written in vector notation as

$$\mathbf{x}^T \cdot \mathbf{y} = [\mathbf{x}_1 \ldots \mathbf{x}_n] \begin{bmatrix} \mathbf{y}_1 \\ \vdots \\ \mathbf{y}_n \end{bmatrix} = [\mathbf{x}_i \mathbf{y}_i + \ldots + \mathbf{x}_n \mathbf{y}_n] \tag{2.13}$$

Some well-known kernels are

Kernel	Description	Hyperparameters
Linear	$K(\mathbf{x}, \mathbf{y}) = \langle \mathbf{x} \cdot \mathbf{y} \rangle$	none
Polynomial	$K(\mathbf{x}, \mathbf{y}) = \langle \mathbf{x} \cdot \mathbf{y} \rangle^d$	d
Radial Basis Function	$K(\mathbf{x}, \mathbf{y}) = \exp(-\gamma \parallel \mathbf{x} - \mathbf{y} \parallel^2)$	γ

(2.14)

A special *cost* hyperparameter C that is kernel-aspecific (in the sense that it is part of the underlying optimization engine deriving the model, and is not part of the actual kernel computations that are performed) relaxes the large margin constraints by allowing support vectors to lie not exactly on a linear margin line. This implements a control mechanism with which the distance of support vectors to the large margin boundary can be more or less penalized. The higher the value of C, the more rigid the boundaries between classes become. Tuning C can have major effects on generalization performance, depending on the kernel that is used.

The *Gram matrix* for a certain kernel $K : X \times X \mapsto \mathbb{R}$ is the matrix $G_{ij} = K(\mathbf{x}, \mathbf{y})$, for all data points $\mathbf{x}, \mathbf{y}$ in the domain of the kernel. This matrix encodes the distances computed by the kernel for a dataset.

For a candidate kernel to be useful, one has to prove that it is *positive definite*: this means its solution is unique and that the optimized problem is convex (see e.g. Schölkopf and Smola [2002]). For convex problems, a local minimum is a global minimum. These problems are easily solvable using linear methods.

2.1.2. PROPOSITION. *Given a kernel* $K : X \times X \mapsto \mathbb{R}$*, with* $\mid X \mid = m$*, let the* $m \times m$ *Gram matrix* $\mathbf{K}$ *consist of all values* $K(\mathbf{x}, \mathbf{y}), \forall \mathbf{x}, \mathbf{y} \in X$*. If* $\mathbf{c}^T\mathbf{K}\mathbf{c} \geq 0$ *for any vector* $\mathbf{c} \in \mathbb{R}^m$*, then* K *is positive definite.*

If the condition $\mathbf{c}^T\mathbf{K}\mathbf{c} \geq 0$ holds only for vectors $\mathbf{c}$ such that $\mathbf{c}^T\mathbf{1}=\mathbf{0}$, K is called *conditionally positive definite* or cpd. The connection between positive definite and conditionally definite kernels is as follows. First, any positive definite kernel is conditionally positive definite (Schölkopf and Smola [2002]). Positive definite kernels are usually interpreted as dot products in feature spaces. Conditionally positive definite kernels can be interpreted as translation-invariant distance measures in feature spaces. Support Vector Machines are translation invariant in the feature space (Zhang et al. [2005]) and therefore can use both types of kernels. The kernel trick, mentioned above, can only be carried out for positive semi-definite kernels: if, for any finite subset[2] $\{\mathbf{x}^1, \ldots, \mathbf{x}^n\}$ of a data space X, and any real numbers $\{c_1, \ldots, c_n\}$

$$\sum_{i,j} K(\mathbf{x}^i, \mathbf{x}^j) c_i c_j \geq 0 \tag{2.15}$$

then there exists a function ϕ such that

$$K(\mathbf{x}, \mathbf{y}) = \phi(\mathbf{x}) \cdot \phi(\mathbf{y}) \tag{2.16}$$

In other words: the kernel can be re-expressed as a linear dot product in feature space. We refer the reader to Schölkopf and Smola [2002] for further details.

As noted above, in contrast to the *eager* Support Vector Machines, *lazy* or memory-based learners defer model construction to the classification stage. The simplest form of memory-based learning is the standard k-nearest neighbor classifier (Cover and Hart [1967]). Depending on the hyperparameter k, for every test point x, a neighborhood of k nearest neighbors in the training data is estimated, using a distance function. The test point receives a class based on majority voting over the classes of the training points in its nearest neighbor set. The nearest neighbor classifier is both a transparent implementation of distance-based classification, and a generic classifier architecture. It can be viewed as a local, kernel-based method that partitions a data space into a number of cells (a *Voronoi tessellation*; Duda et al. [2001]). Its error rate has an upper bound of less than twice the *Bayes error rate* (Duda et al. [2001]),

[2]Here, $\mathbf{x}^i$ refers to the i-th element of X'.

which is a statistical lower bound on the classification error for a certain task given a set of features. The use of local context can be traced back to many machine learning algorithms, e.g. decision trees (Quinlan [1993]), and some support vector kernels (e.g. the Radial Basis Function kernel; see Jebara [2003]). The relationship between nearest neighbor classification and Support Vector Machines is investigated in e.g. Decoste and Mazzoni [2003], who propose a classification technique based on *nearest support vectors*, and Zhang et al. [2006], who propose a hybrid SVM/k-nearest neighbor machine learning algorithm.

Memory-based learning has proved quite successful for natural language analysis (see e.g. Daelemans and van den Bosch [2005] and Hendrickx [2005]). One explanation for this success is the fact that language displays *pockets of exceptions*: small families of subregularities, which, while easily 'compiled away' by model-based methods, are fully retained in memory by memory-based methods (Daelemans et al. [1999]).

2.1.1 Bias, variance and noise

The error of a classifier is often decomposed into bias error, variance error, and noise (Breiman [1996]): the bias error is the systematic, intrinsic error of a classifier, the variance error is its data dependent error, and noise corresponds to errors (either in features or classes) in the data. Specifically, the expected zero-one loss of a classifier C is measured as

$$E(C) = \sum_x p(x)(\sigma_x^2 + bias_x^2 + variance_x) \tag{2.17}$$

with σ_x^2 the noise in the data for datum x. Noise is often assumed to be zero (Kohavi and Wolpert [1996]) as reliably estimating noise is often infeasible for large datasets. Kohavi and Wolpert [1996] propose the following definition of bias error and variance error:

$$bias_x^2 = \frac{1}{2} \sum_{y \in Y} \left[P_{Y,X}(Y = y \mid X = x) - P_{\mathcal{T}}(\mathcal{L}(\mathcal{T})(x) = y \right]^2 \tag{2.18}$$

$$variance_x = \frac{1}{2} \left(1 - \sum_{y \in Y} P_{\mathcal{T}}(\mathcal{L}(\mathcal{T})(x) = y)^2 \right) \tag{2.19}$$

According to this definition, the bias error of a classifier at a data point x is the squared difference between the true class observed for x in the training data X, Y (X a feature space and Y a class space), and the class predicted for x in the hypothesis space $\mathcal{T}$, i.e. the output of the classifier $\mathcal{L}$ trained on $\mathcal{T}$ and applied to x. In Kohavi and Wolpert [1996], bias and variance error are measured on the basis of the following procedure: each data set is partitioned into a training set d and a test set t. The training data d is partitioned into 50 training sets

of size $2m$, where m is 100 for data sets less than 1,000 data points, and 250 otherwise. The bias and variance error estimates are then derived from training the classifier on the 50 training subsets in turn, and applying it to the test data t.

2.2 Hyperparameter estimation

We noted in Section 2.1 that machine learning algorithms are complex, parameterized functions of which the hyperparameters need adequate estimation. In this section, we outline a procedure for estimating these hyperparameters. In Section 4, we will use this procedure in a number of experiments.

For a given machine learning algorithm and an evaluation function - typically an empirical loss function, like accuracy - one needs to simultaneously optimize all hyperparameters of the learning algorithm such that the loss function is minimized after training. Hyperparameter values need to be robust and should lead to adequate results of the trained classifier when applied to new, unseen cases. While sometimes default settings can provide adequate results, there is no guarantee this will in general lead to acceptable performance. Daelemans et al. [2003] provide evidence for the fact that variation in accuracy arising from hyperparameter settings and interaction with information sources (such as arising from feature selection) is often higher than variation between separate machine learning algorithms using default settings. This implies that the widely used methodology of comparing classifiers with default hyperparameter settings is unreliable, and that, in order to faithfully compare classifiers on a given task, one needs to optimize hyperparameters for both classifiers.

Hyperparameter spaces are usually extremely sparse, and search in this event space is not transparently conditioned on an intermediate search result, which makes this a very hard search problem. Optimal combinations of hyperparameter values can be quite rare. Proposed methods for hyperparameter estimation usually view the problem as a search problem. The `paramsearch` estimation procedure of van den Bosch [2004] is an example of a *grid search* algorithm; on the basis of a sweep across a discrete partitioning of the hyperparameter space, it progressively builds bigger training/test partitionings for successful hyperparameter settings, in order to cope with the problem of accidentally discarding settings that perform badly on small data sets, but better on bigger sets. It essentially optimizes generalization accuracy by performing cross-validation for small datasets (<1000 points) and progressive sampling for bigger datasets.

Friedrichs and Igel [2005] tune SVM hyperparameters using an evolutionary technique (*covariance matrix adaptation evolution*). Their approach is related to ours in the sense that they optimize a fitness function that directly is associated with a possibly non-differentiable empirical loss function. Their method also has hyperparameters of its own, which are set to default settings or to heuristically

determined values.

Gradient minimization approaches, such as Bengio [2000] and Chapelle et al. [2000] work by minimizing a cost function and performing search in the hyperparameter space based on the gradient of this cost function with respect to the kernel hyperparameters. This implies that the cost function needs to be continuously differentiable for these hyperparameters, which is a strong assumption, and rules out many reasonable cost functions.

Many optimization problems involve reliable estimates of the probability of rare but interesting events. The Cross-Entropy method of Rubinstein [1999], explained below, is a case in point; using a technique called *importance sampling*, rare but important events are made more likely to occur in a certain sequence of observations, such that reliable estimates of the probabilities for these events can be derived. This process can be parameterized, and if one can devise an isomorphism between the desired outcome of the sampling process and the parameters guiding this process, one effectively has a search algorithm in a rare event space. It is possible to treat an optimal hyperparameter setting as a rare event, and casting it into the framework of the cross-entropy method leads to a novel way of finding adequate hyperparameter settings. We propose in this section a beam search algorithm inspired by the cross-entropy method. First, we will discuss the cross-entropy method in some detail.

2.2.1 The Cross-Entropy Method

The Cross-Entropy (CE) Method (Rubinstein [1999]) is a stochastic optimization method that iteratively performs an importance-based sampling on an event space using a family of parameterized sampling principles. It has been successfully applied to routing problems (Cadre [2005]. Chepuri and de Mello [2005]), network optimization (de Boer [2000]) and HMM parameter estimation (Dambreville [2007]). An excellent introduction can be found in de Boer et al. [2005].

The CE-method in its most general form can be described as follows. Suppose we want to find a certain $\hat{p} = \text{argmax}_p\, S(p)$, where S is a general empirical loss function, which is not necessarily continuous. Let $\mathcal{X}$ be a vector space[3] consisting of vectors (events, in the nomenclature of cross-entropy methods) $X_i, \ldots X_n$, with every vector $X_i \in \mathbb{R}^n$ for some positive integer n. We denote the j-th element of X_i with X_{ij}. Let $f(\cdot; u)$ be a probability density function on $\mathcal{X}$ parameterized by u. Let's assume we have an *importance measure*, an independent evaluation function, such that we can evaluate for every event how important this event is. We would like to adapt the probability of rare events proportional to their importance. To estimate whether event X_i is important, i.e. a

[3] In order to be compliant with the standard notation used in the Cross-Entropy Method literature, we will use a different notation for vectors and their components, as described in the text.

candidate solution to the optimization problem, we need to compute the chance that $S(X_i) \geq \gamma$, where γ is a real number. If this chance is very low, then X_i is a rare event. Let the indicator function $I_{\{S(x)\geq\gamma\}}$ be a binary valued function over the set $\{X_i \mid S(X_i) \geq \gamma\}$; $I_{\{S(x)\geq\gamma\}}(x) = 1\ iff\ x \in \{X_i \mid S(X_i) \geq \gamma\};\ 0\ else$. For a randomly drawn sample $X_1, \ldots X_n$, the maximum-likelihood estimate of this chance would be

$$\frac{1}{N}\sum_{i=1}^{N} I_{\{S(X_i)\geq\gamma\}} \tag{2.20}$$

For rare events, reliable random sampling is hard to perform and large samples need to be drawn. If we were to have a 'better' estimate of f, say g, a pdf based on importance sampling, the chance $P(S(X_i) \geq \gamma)$ can be estimated with more confidence by

$$\frac{1}{N}\sum_{i=1}^{N} I_{\{S(X_i)\geq\gamma\}} \frac{f(X_i;u)}{g(X_i)}, \tag{2.21}$$

However, g is based on importance sampling, and thus depends on the very probability we want to compute. The idea now is to stepwise approximate g by a sequence of distributions $g_1, \ldots, g_n$ such that the Kullback-Leibler (KL)-distance between every g_i and $f(X_i;u)$ is minimized. Every such distribution would maximize the probability of the samples with respect to γ. The CE-method repeatedly performs importance sampling, and adjusts the parameter vector u such that the KL-distance between the importance sample g and the current pdf f conditioned on u is minimized.

Minimizing the KL-distance between two pdfs g and $f(x;u)$

$$KL(g,f) = \int g(x)\ln g(x)dx - \int g(x)\ln f(x)dx \tag{2.22}$$

amounts in the case of f parameterized by parameter vector u to solving

$$\max_v \int g(x)\ln f(x;v)dx \tag{2.23}$$

By defining a *change of measure* as a likelihood ratio (w an arbitrary parameter vector)

$$W(X_i;u,w) = \frac{f(X_i;u)}{f(X_i;w)} = \\ \exp\left(-\sum_{j=1}^{N} X_{ij}(\frac{1}{u_j} - \frac{1}{v_j})\right)\prod_{j=1}^{N}\frac{v_j}{u_j} \tag{2.24}$$

the adapted parameter vector $\hat{v}_t$ (t a time index) can be derived as follows:

$$\hat{v}_t = \max_v \frac{1}{N}\sum_{i=1}^{N} I_{\{S(X_i)\geq\gamma\}} W(X_i;u,w)\ln f(X_i;v) \tag{2.25}$$

with $X_1, \ldots, X_n$ a random sample drawn from $f(\cdot; w)$. This, through some basic differentiation, can be brought into the following parameter update formula

$$\hat{v}_t = \frac{\sum_{i=1}^{n} I_{\{S(X_i) \geq \gamma_t\}} W(X_i; u, \hat{v}_{t-1}) X_{ij}}{\sum_{i=1}^{n} I_{\{S(X_i) \geq \gamma_t\}} W(X_i; u, \hat{v}_{t-1})} \tag{2.26}$$

for the details of which we refer the reader to de Boer et al. [2005]. The general CE algorithm for rare event simulation is outlined in Algorithm C.1 in Appendix C.

2.2.2 Elitist Cross-Entropy Hyperparameter Estimation

It is quite tempting to apply the CE-method to the problem of hyperparameter estimation. A large body of empirical and analytical work demonstrates that the CE-method is efficient and robust against local minima (see de Boer et al. [2005] for an overview). In this section, we recast the hyperparameter estimation problem into the CE framework and demonstrate that a beam search algorithm inspired by the CE-method indeed can be applied, provided some facilities are added to the basis algorithm.

2.2.3 Change of measure

For the essential ingredients of the CE-method to be applicable to the problem of hyperparameter estimation, two things are necessary: first, the sampling of candidate hyperparameter settings must be made dependent on the parameter vector updated by the CE algorithm, and secondly, a suitable change of measure guiding the search process must be devised. Since initialization of the hyperparameter search process usually is arbitrary - based on a random vector, eventually with values restricted to a certain range - the likelihood term W (2.24) would not make sense. But leaving it out of the parameter update formula (2.26) would reduce the search process to crude trial-and-error Monte-Carlo search (de Boer et al. [2005]).

Genetic algorithms control the direction of search in non-probabilistic spaces using history mechanisms such as elitism (de Jong [1975]): explicitly favoring the reproduction of an elite of fit individuals. In the present case, the elitist solution at time t would consist of the best performing hyperparameter vector encountered. The information in this successful elitist solution seems valuable for steering the search process to more optimal solutions. Let us refer to the elitist solution found at time t with E^t. In order to evaluate whether a candidate solution differs significantly from the elitist solution, Euclidean distance is a natural distance measure. The following normalized Euclidean distance, taking values in the interval [0,1], can be used as a change of measure for the CE algorithm; just like the original log-likelihood ratio in (2.24) produces a value of 1 whenever the two pdfs $f(X_i; u)$ and $f(X_i; w)$ are the same, this measure

produces 1 whenever the Euclidean distance between X_i and E^t is zero.

$$W(X_i^t; E^t) = 1 - \frac{\sqrt{\sum_{j=1}^{m}(X_{ij}^t - E_j^t)^2}}{\sqrt{\sum_{j=1}^{m}(X_{ij}^t)^2}\sqrt{\sum_{j=1}^{m}(E_j^t)^2}} \tag{2.27}$$

2.2.4 Conditional drawing

In order to condition the drawing at time t of a random sample $X_1, \ldots, X_n$ from the hyperparameter event space $\mathcal{X}$ on the parameter vector derived at time t, we limit the choice for every X_{ij} to lie within the interval

$$[\hat{v}_{t,j} * (1 - \mu), \hat{v}_{t,j} * (1 + \mu)] \tag{2.28}$$

where μ is a width parameter. This is essentially a beam search operation: search is carried out in a limited region of the search space, constrained by the width of the beam constituted by the parameter μ.

In order to avoid the algorithm getting stuck in a local minimum, we run it in parallel in a number of n independent threads. From these threads, the maximum result is the winner. Algorithm C.2 lists the final hyperparameter algorithm. Notice that whenever $W(X_i^t; E^t)$ evaluates to 1, the parameter adaptation step reduces to crude Monte Carlo search. The final parameter vector v^t is the solution to the hyperparameter estimation problem. A typical stopping condition would be a persistent γ for a number of iterations.

Solving a hyperparameter estimation problem with a method that itself has hyperparameters may seem odd and circular. Yet, this is common as optimization algorithms are seldom parameter-free; compare for instance Friedrichs and Igel [2005] who use parameterized evolutionary techniques to optimize support vector machines. The hyperparameters of the CE algorithm are known to predominantly influence the speed of convergence rather than the quality of the exact solution. de Boer et al. [2005] note that these hyperparameters can be easily found using an adaptive algorithm. Another option is to apply the CE-method iteratively to itself, migrating from relatively large hyperparameter spaces to ever smaller hyperparameter spaces. A suitable metaphor would be to adjust a device with analogue controls by means of a superimposed stack of devices with digital, discrete controls, where device t adjusts device $t-1$, and t has larger intervals between the digits (less 'clicks') than device $t-1$. Put differently, the hyperparameters of the CE algorithm are piecewise functions, whereas the hyperparameters of the procedure to be optimized are continuous functions. We shall not pursue this issue further in this chapter, but we will demonstrate the relative insensitivity of the ECE-method to various choices for N, ρ, μ in a separate experiment, described below.

The ECE algorithm is completely neutral with respect to the evaluation function S, which can consist of any machine learning algorithm, eventually wrapped in extensive cross-validation procedures.

2.2.5 Experiments

We started comparing ECE with `paramsearch` on single test data splits; subsequently, we compared the ECE solutions with the `paramsearch` solutions using 10-fold cross-validation in order to assess the generalization capabilities of the ECE solutions. Finally, we used the ECE algorithm to optimize a challenging loss function based on a window-based recall and precision-measure for the task of topic segmentation of transcribed speech. Throughout we used the SVMlight software[4].

2.2.6 Accuracy-based optimization

Our data consisted of 11 datasets, originating from the UCI repository (Hettich and Bay [1999]), the LIBSVM dataset repository (Chang and Lin [2001]), and the SVMlight example data collection (Joachims [2004]). None of the datasets used was published with a separate development set for hyperparameter optimization, and some did not have a separate test set either. Notice that we are only interested in *relative* performance of hyperparameter optimization techniques, and not in *absolute* results obtained on this data. For every dataset listed in Table 2.2.6 the experimental procedure was the following. First, we manually split the concatenation of training and test data into a training part, a test part and a development part. For `ijcnn1` and `a3a` we used the standard test sets provided by the publishers of this data, in order to evaluate performance on very large test sets. The development part was subsequently split into a development train and development test part. The development set as a whole was used by `paramsearch` to derive the SVM parameters. The training/test parts of the development set were used by the ECE algorithm as follows: for every candidate hyperparameter vector, training on the basis of this hyperparameter vector took place on the development training set, and testing on the development test set. In theory this could set back the ECE algorithm compared to `paramsearch`, as the latter applies cross-validation and progressive sampling to the entire development set, whereas the ECE algorithm only uses one fixed partitioning of the development data. While the ECE algorithm can be easily furnished with similar cross-validation routines, as it is completely neutral with respect to the performance function, we did not implement this, and report results on the basis of the fixed training/test splits of the development data. Second, we applied 10-fold cross-validation to the training splits we created,

[4] Available from `http://svmlight.joachims.org/`.

and applied Wilcoxon's Rank Sum test to the results, in order to assess the differences between `paramsearch` and ECE.

The hyperparameter search space can optionally be constrained by imposing limits on the range of generated random values. For instance, degrees larger than 5 for a polynomial kernel are in practice rather unlikely, as well as values exceeding 1000 for the SVM regularization parameter C. This domain knowledge, although not crucial to the performance of the ECE algorithm, can be easily applied if reliable. The (uniform) limits we imposed on the various SVM parameters are listed in Table 2.2.6; they address the regularization parameter `C`, the RBF kernel parameter `g`, the positive class penalty term `j`, and the polynomial degree parameter `d` (see Joachims [2004] for details on these parameters).

Task	Features	devset-train	devset-test	train	test
svmlight ex1	sparse	400	300	1000	300
news20	1,355,191 (sparse)	1500	319	15,000	3000
svmguide1	21	700	389	1500	500
a1a	123 (sparse)	200	105	1000	300
a3a	123 (sparse)	700	189	2000	29,376
a6a	123 (sparse)	1000	259	8000	1500
ijcnn1	22	10,000	4990	35,000	91,701
ringnorm	20	1500	336	300	5000
splice	60	300	100	600	2175
rcv1	47,236 (sparse)	1242	358	15,000	3000
diabetes	8	100	68	500	100

Table 2.1: Number of features and examples of the 11 datasets.

SVMlight kernel parameter	Limit
-g	$0.01 \leq g \leq 10$
-j	$0.1 \leq j \leq 5$
-c	$1 \leq c \leq 1000$
-d	$1 \leq d \leq 5$

Table 2.2: Uniform limits imposed on the value range of SVM kernel parameters.

We let the `paramsearch` algorithm determine the kernel type (RBF or polynomial), and fixed this parameter when running the ECE algorithm, in order to perform a close 'within-kernel' comparison of results. For every run of the ECE algorithm, we used 'common-sense' settings of $N = 10, \rho = 0.1, \mu = 0.9$; we discuss this in more detail in Section 2.2.8. We compared both `paramsearch` and ECE with a baseline: default parameter settings for the kernel-type determined by `paramsearch`. First, we evaluated the hyperparameter values on the

Dataset	Fixed train/test split (accuracy)			10-fold CV average accuracy		
	Default	paramsearch	ECE	paramsearch	ECE	WRS
svmlight example1	94.33	94.33	94.34	97.19	96.89	=
news20	95.8	95.63	96.77	95.2	96.75	+
svmguide1	94	85	85	76.26	68.6	=
a1a	79	87	99.3	80.2	80.2	=
a3a	75.94	81.45	81.13	80.4	79	=
a6a	82.8	83.07	77.7	83.8	82.38	=
ijcnn1	98.18	97.47	98.52	97.45	98.21	=
ringnorm	50.78	93.82	98.34	90.66	95.66	+
splice	52.18	87.08	89.1	80.8	84.5	=
rcv1	97	96.7	97.13	96.84	97.15	=
diabetes	86	85	87	71.8	73.2	=

Table 2.3: Experimental results, for a fixed training/test split, and Wilcoxon's Rank Sum (WRS) test applied to 10-fold cross-validation results obtained on the training data; '+' indicates ECE outperforming `paramsearch` according to the WRS test; '=' indicates equivalence.

fixed training/test splits. Subsequently, we applied 10-fold cross-validation to the training data, after which we compared the accuracy scores for ECE and `paramsearch` using Wilcoxon's Rank Sum test. As can be seen in Table 2.2.6, for the fixed training/test data split, in 10 out of 11 cases, ECE finds hyperparameter settings that are as good as or better than `paramsearch`, even though no cross-validation was used. The baseline of default hyperparameter settings appears acceptable in 7 out of 11 cases, which means one cannot rely on default settings in general, as noted above. The cross-validation results show that ECE finds equivalent and robust hyperparameter values compared to `paramsearch`. Again, this is remarkable, as ECE was applied to a single data split.

In practice, the ECE algorithm proved very efficient, with convergence under 100 iterations, in a matter of minutes[5]. The `paramsearch` algorithm ran significantly slower, sometimes taking several hours to finish.

2.2.7 Optimization for skewed class distributions

We now turn from accuracy-based optimization to an even more challenging problem: optimization of a loss function for skewed class distributions. When class distributions display a high level of entropy, i.e. $P(c_i \mid T) \approx P(c_j \mid T), i \neq j$ for any two classes c and training data T, accuracy is an acceptable measure of quality for a classifier. But when class distributions are highly skewed, recall, precision and harmonic means of these such as the F1-score are better measures.

[5] On a 2G RAM 2.0 GHz Pentium dual-core OSX machine.

Topic segmentation, segmenting a text into separate topics, is a typically class-imbalanced task. The number of linguistic units on which segmentation is based (such as sentences) typically by far exceeds the number of actual topics. Consequently, optimizing a classifier for accuracy would automatically favor a majority classifier that labels all sentences as not opening a topic. Optimization for the classical notions of recall and precision would not work well here either: for instance, a topic segmenter that always predicts a topic boundary close but not exactly corresponding to the ground truth prediction would produce zero recall and precision, while its performance can actually be quite good.

Specific measures such as P_k and WindowDiff (Pevzner and Hearst [2002]) compute recall and precision in a fixed-size window to alleviate this problem, but they do not penalize false negatives and false positives in the same way. For topic segmentation, false negatives probably should be treated on a par with false positives, to avoid undersegmentation. To this end, Georgescul et al. [2006] proposed a new, cost-based metric called Pr_{error}:

$$Pr_{error} = C_{miss} \cdot Pr_{miss} + C_{fa} \cdot Pr_{fa} \tag{2.29}$$

Here, C_{miss} and C_{fa} are cost terms for false negatives and false alarms; Pr_{miss} is the probability that a predicted segmentation contains less boundaries than the ground truth segmentation in a certain interval of linguistic units (such as words); Pr_{fa} denotes the probability that the predicted segmentation in a given interval contains less boundaries than the ground truth segmentation. We refer the reader to Georgescul et al. [2006] for further details and the exact computation of these probabilities.

The ECE algorithm was applied to multimodal, topic-segmented meeting data from the AMI project (Carletta et al. [2005]), consisting of 50 videotaped, manually transcribed scenario-based meetings annotated for main topic structure. From this data, a mixture of phonetic and lexical features was extracted. Lexical cues for topic openings were determined using χ-square-based term extraction. The LCSEG algorithm proposed by Galley et al. [2003] was used to produce lexical cohesion scores and cohesion probabilities between blocks of words. Further, speaker activity change in a window of 5 seconds was measured, as well as the number of pauses and the amount of word repetition. These features are similar to those proposed in Galley et al. [2003]. The linguistic unit on which segmentation is based was the *spurt*: consecutive speech in which pauses between utterances are under 5 seconds.

As the `paramsearch` algorithm does not cater for minimizing a loss function of this type, we compared the results of the ECE algorithm applied to optimization of the Pr_{error} measure to four different baselines: A- (generating negative classes only), A+ (generating positive classes only), R (generating random positive and negative classes) and R-n (generating random classes, with n positive classes, where n is the number of known segments in the test data). As it turns out, both the A-, A+, and R baselines yielded similar performance

(errors around 0.5). This indicates that a 0.5 error rate and the 0.48 producing R-n baseline are the only non-trivial baselines. We set C_{miss} and C_{fa} to 0.5, in order to penalize undersegmentation as much as oversegmentation.

Results are listed in Table 2.2. The ECE algorithm appears to significantly outperform the 4 baselines, and produces scores comparable to the ones reported by Georgescul et al. [2006] on ICSI meeting data.

Fold	Pr_{error}	A-	A+	R	R-n
1	34.95	0.5	0.5	0.5	0.48
2	31.98	0.5	0.5	0.498	0.499
3	38.87	0.5	0.5	0.498	0.49
4	37.69	0.5	0.5	0.498	0.483
5	36.47	0.5	0.5	0.5	0.47
6	35.93	0.5	0.5	0.499	0.51
7	33.85	0.5	0.5	0.5	0.489
8	38.62	0.5	0.5	0.499	0.486
9	35.92	0.5	0.5	0.498	0.499
10	35.13	0.5	0.5	0.497	0.485
Average	35.9	0.5	0.5	0.499	0.484

Figure 2.2: Topic segmentation results.

2.2.8 Persistence of results

In a separate experiment, we evaluated the persistence of results of the ECE algorithm by varying its hyperparameters. We took the `ringnorm` dataset, and varied N, ρ, μ ($N \in \{5, 10, 50\}$, $\rho \in \{0.1, 0.5, 0.7\}$, $\mu \in \{0.1, 0.5, 0.9\}$), training on the training part of the development data, and testing on the corresponding test partition. The stopping criterion for the ECE algorithm consisted of a persistent γ for 50 iterations. Figure 2.3 illustrates the convergence rates for the various combinations of the ECE hyperparameter values and $N \in \{5, 10\}$. As can be seen, all variants converge to the same level of accuracy (i.e. >99%), demonstrating persistence of accuracy. When $N = 10$, the number of iterations prior to convergence goes up significantly. For the cases where $N = 50$, we only obtained steadily increasing accuracy in 3 cases under 50 iterations; this demonstrates that the number of iterations prior to convergence is at least proportional to the sample size.

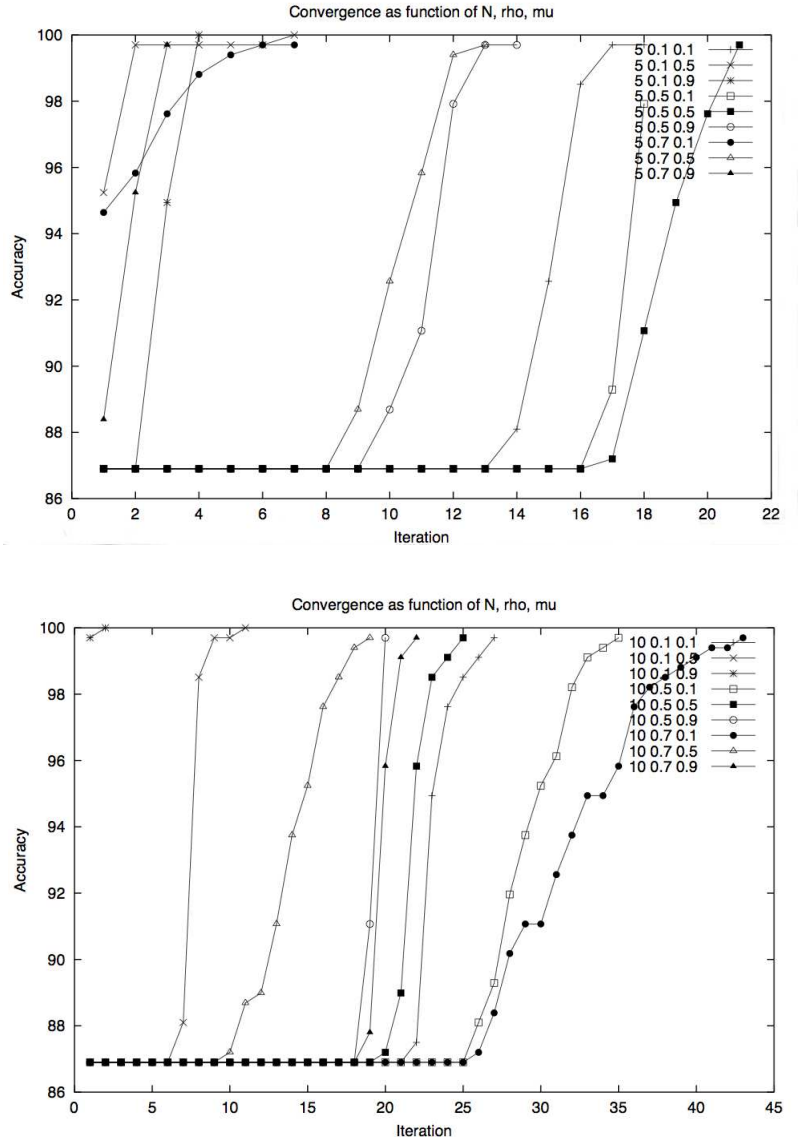


Figure 2.3: Iteration/accuracy trade-offs of ECE for the `ringnorm` dataset. Values of respectively N, ρ, μ are listed in the legend. Accuracy was measured on the test part of the development data.

2.3 Vector space models of documents

As the work in this thesis is heavily dependent on vector space models, we will provide in this section an introduction to the basic concepts.

Ever since the seminal work on document representation by Luhn [1957] and Salton et al. [1975], vector space models of documents in information retrieval and machine learning have become ubiquitous. Vector representations of documents usually consist of statistical scores for a certain index vocabulary, binary *on/off* representations signaling the presence of absence of a certain word, or normalized frequencies. For instance, given a toy vocabulary consisting of pairs of index terms and their index

$$\{(1, hate), (2, like), (3, good), (4, awesome), (5, bad), (6, poor), (7, really)\} \tag{2.30}$$

one could index documents for sentiment. Every document becomes represented by a vector, in which every position is bound to a certain word, and every value for that position consists of, say, the frequency of that word in the document:

$$\begin{aligned} &I\ really\ really\ like\ this\ movie,\ the\ acting\ is\ awesome \quad \mapsto \\ &< 0, 1, 0, 1, 0, 0, 2 > \end{aligned} \tag{2.31}$$

So, the vector space model basically is a *bag-of-words* model: it treats documents as multisets (bags) of words: unordered collections with repetition, and represents documents as ordered vectors consisting of counts of designated index words.

Usually, the terms to index a document with are selected through term selection. For instance, the $tf.idf$ representation is widespread, and combines term frequency (tf, the frequency of a term t in a specific document d) and inverse document frequency (idf, the number of documents in a collection D that contain a specific term) in one complex measure that can be used to rank terms for importance:

$$\begin{aligned} tf(t, d) &= \frac{|t \in D|}{\sum_i |t_i \in d|} \\ idf(t, D) &= \log \frac{|D|}{|d \in D : t \in d|} \\ tf.idf(t, d, D) &= tf(t, d) \times idf(t, D) \end{aligned} \tag{2.32}$$

The $tf.idf$ score is dominated by the idf-score, which will be low when the number of documents containing a specific term will approach the total number of documents. This will demote words that occur in many documents (such as articles, prepositions, etc.), and promote words that are distinctive. Usually, the idf scores are computed from a training corpus, and the tf-scores are taken from a test document to be indexed.

Given a suitable vector representation of documents, similarity between documents can be measured by operations defined on vectors, for instance the dot product (expressed here as a *linear kernel* on two vectors $\mathbf{x}$ and $\mathbf{y}$):

$$K_{LIN}(\mathbf{x}, \mathbf{y}) = \sum_i \mathbf{x}_i \mathbf{y}_i \tag{2.33}$$

or by measuring the angle between two vectors $\mathbf{x}$ and $\mathbf{y}$, which relates document length ($\mid \mathbf{x} \mid$, $\mid \mathbf{y} \mid$) and dot products:

$$\begin{aligned} \mid \mathbf{x} \mid &= \sqrt{\mathbf{x}\mathbf{x}} \\ \angle \mathbf{x}\mathbf{y} &= \arccos\left(\frac{K_{LIN}(\mathbf{x},\mathbf{y})}{|\mathbf{x}||\mathbf{y}|}\right) \end{aligned} \tag{2.34}$$

As noted above, the well-known Euclidean distance measure is

$$K_{EUCLID}(\mathbf{x}, \mathbf{y}) = \sqrt{\sum_i (\mathbf{x}_i - \mathbf{y}_i)^2} \tag{2.35}$$

Alternatively, similarity between document vectors can be measured as the cosine between their representative, length-normalized vectors:

$$\cos(\mathbf{x}, \mathbf{y}) = \frac{\mathbf{x} \cdot \mathbf{y}}{\mid \mathbf{x} \mid \cdot \mid \mathbf{x} \mid} \tag{2.36}$$

where

$$\mid \mathbf{x} \mid = (\mathbf{x} \cdot \mathbf{x})^{1/2} (\text{the Euclidean norm of } \mathbf{x}) \tag{2.37}$$

The use of these essentially Euclidean distance measures for document classification has a long and thriving tradition (see e.g. Joachims [2002]). Many variations have been proposed that address different normalizations of vectors. For instance, given a document representation consisting of vectors of $tf.idf$ values, the use of L2-normalization of these values in combination with a linear kernel has been known to produce very accurate results, e.g. Zhang et al. [2005].

$$\textit{Given a vector } \mathbf{x}, \textit{ its } L2 - \textit{normalized variant is } \left(\frac{\mathbf{x}_i^2}{\sum_{i=1}^{n} \mathbf{x}_i^2}, \ldots, \frac{\mathbf{x}_n^2}{\sum_{i=1}^{n} \mathbf{x}_i^2} \right). \tag{2.38}$$

The underlying assumption of the Euclidean approach is that document space is flat, without curvature, and that distances between documents are measured along straight lines.

2.4 The multinomial simplex

In this section, we introduce the key concepts and formal methods behind the classification framework that is central to our work. Recent work by Lebanon [2005] states that the vector space model for documents, although useful, is an ad hoc approximation, and that documents are more naturally represented in the multinomial manifold: a curved information structure arising from so-called L(evel)1-embeddings. A manifold is a topological space that is (only) locally Euclidean. Geometric objects such as curves and spheres are manifolds, as they possess Euclidean structure in lower dimensions (like a sphere projected onto 2D). An infinitely differentiable manifold is called a Riemannian manifold when equipped with a distance metric measuring the distance between two arbitrary points.

The *multinomial simplex*[6] is the parameter space $\mathbb{P}^n$ of the multinomial distribution equipped with the so-called Fisher information metric (Lafferty and Lebanon [2005]):

$$\mathbb{P}^n = \left\{ \mathbf{x} \in \mathbb{R}^{n+1} : \forall j \ \mathbf{x}_j \geq 0, \sum_{i=1}^{n+1} \mathbf{x}_i = 1 \right\} \tag{2.39}$$

Every $\mathbf{x} \in \mathbb{P}^n$ is a vector of $n+1$ probabilities, or outcomes of an experiment. The analogy with normalized word frequencies in a document is the following: every word is an experiment, and its normalized frequency in the document (the number of times the word occurs, divided by the total number of words) is its outcome, which corresponds to L1-normalization. Representing a document as a vector $\mathbf{x}$ of word frequencies, we note that

$$\textit{Given a vector } \mathbf{x}, \textit{ its } L1-\textit{normalized variant is } \left(\frac{\mathbf{x}_i}{\sum_{i=1}^{n} \mathbf{x}_i}, \ldots, \frac{\mathbf{x}_n}{\sum_{i=1}^{n} \mathbf{x}_i} \right) \tag{2.40}$$

A multinomial distribution is a probability distribution consisting of n separate independent trials in each of which a set of random variables $X_i \ldots X_k$ was observed with probabilities $p_i \ldots p_k$ such that $\sum_{i=1}^{k} p_i = 1$. For any document, the random variables correspond to the different words occurring in it, and the probabilities p_i are the L1-normalized frequencies of those very words in that particular document. The connection of L1-normalization with the multinomial distribution explains the 'multinomial' epithet of this approach. The multinomial manifold therefore is a natural habitat for document representations under L1-normalization.

[6] A simplex is a manifold with boundaries and corners (Lee [1997]); see also Appendix A.

The native metric on the multinomial simplex is the Fisher Information[7]:

$$g_\theta(u,v) = \sum_{i=1}^{n+1} \frac{u_i v_i}{\theta_i} \tag{2.41}$$

with $\theta \in \mathbb{P}^n$ and $u, v \in T_\theta \mathbb{P}^n$ are vectors tangent to θ in the space $\mathbb{P}^n$.

As it turns out, there exists a diffeomorphism that relates $\mathbb{P}^n$ to the positive n-dimensional *sphere* $\mathcal{S}_+^n$

$$\mathcal{S}_+^n = \left\{ \; x \in \mathbb{R}^{n+1} : \forall j \; x_j \geq 0, \sum_{i=1}^{n+1} \mathbf{x}_i^2 = 1 \; \right\} \tag{2.42}$$

namely

$$F(x) = (\sqrt{\mathbf{x}_1}, \ldots, \sqrt{\mathbf{x}_{n+1}}) \tag{2.43}$$

This is a *pullback* (see Appendix A): it pulls back the distances measured on the n-sphere onto the multinomial simplex. It allows for measuring the geodesic distance between points $\mathbf{x}, \mathbf{y}$ in $\mathbb{P}^n$ by measuring the distance between $F(\mathbf{x})$ and $F(\mathbf{y})$ on $\mathcal{S}_+^n$, which are connected by a curve, the shortest path that actually is a segment of a great circle:

$$D(\mathbf{x}, \mathbf{y}) = \arccos \left(\sum_{i=1}^{n+1} \sqrt{\mathbf{x}_i \mathbf{y}_i} \right) \tag{2.44}$$

Thus, distances between objects (such as documents) in the multinomial space are measured taking into account the intrinsic curvature of the lines connecting them. This sets the multinomial approach apart from the Euclidean approach, where shortest distance is computed irrespective of curvature. We shall refer to distance measures on the multinomial simplex as geodesic kernels. The reader is referred to Kass [1989] for further details.

An L_p^d-normalization formally corresponds to an embedding of data into the space $\mathbb{R}^d$ endowed with the L_p norm $\| \; x \; \|_p$. This means that performing the normalization to a certain datum automatically embeds the datum into the corresponding information space. Usually, the d superscript is dropped. For any $\mathbf{x} \in \mathbb{R}^d$,

$$\| \; \mathbf{x} \; \|_p = \left(\sum_{i=1}^{d} | \; \mathbf{x}_i \; |^p \right)^{1/p} \tag{2.45}$$

When we set $p = 1$, the L_1 norm gives rise to the Manhattan distance, when measuring the distance between two points $\mathbf{x}, \mathbf{y} \in \mathbb{R}^d$:

$$MHD(\mathbf{x}, \mathbf{y}) = \sum_{i=1}^{d} | \; \mathbf{x}_i - \mathbf{y}_i \; | \tag{2.46}$$

[7]Sometimes, we will write vectors in non-bold, whenever this is appropriate according to convention.

2.5 Geodesic kernels

Recently, the use of kernels operating on probabilistic representations of documents have become quite popular. Moreno et al. [2003] propose the use of Kullback-Leibler divergence-based kernels for speaker identification and image classification. This type of kernel has found its way into text classification as well (e.g. Coutinho and Figueiredo [2005]). Likewise, Jebara et al. [2004] investigate a probability product kernel based on products of L2-normalized probabilities. This work demonstrates a growing awareness of the benefits of combining generative models (e.g. language models) with discriminative models (supervised learning), as already proposed in Jebara [2001].

2.5.1 The Information Diffusion Kernel

Lafferty and Lebanon [2005] and Lebanon [2006b] show that applying geodesic kernels to document classification tasks produces state-of-the-art results. In particular, Lafferty and Lebanon [2005] propose the following *information diffusion kernel* on the multinomial manifold:

$$K_t^{ID}(\mathbf{x},\mathbf{y}) = (4\pi t)^{-\frac{n}{2}} \exp\left(-\tfrac{1}{t} \arccos^2\left(\sum_{i=1}^{n} \sqrt{\mathbf{x}_i \mathbf{y}_i} \right) \right) \tag{2.47}$$

This is a one-parameter kernel, n being the dimension of the data. For sufficiently small $t \in [0, \epsilon)$, this kernel is positive definite, guaranteeing a unique solution to the convex problem the kernel machine has to solve (Lafferty and Lebanon [2005]).

Interestingly, the so-called Bhattacharyya distance occurs as a subterm in this kernel:

$$B(p,q) = \sum_{x \in X} \sqrt{p(x)q(x)} \tag{2.48}$$

with p, q two probability distributions over the same event space X. The Bhattacharyya distance is basically Euclidean distance restricted to probability distributions. In kernel (2.47), the inverse cosine (arccos) is used to measure distance across curves, and the Bhattacharyya kernel introduces a local notion of Euclidean distance. The Information Diffusion Kernel is related to the *heat kernel* (Lafferty and Lebanon [2005]), which plays a central part in the modeling of spreading activity through networks, in both physics (e.g. temperature flow) and information science.

The following two-hyperparameter (n, t) variant of the K^{ID} kernel has an additional scaling facility:

$$K_{n,t}^{IDS}(\mathbf{x},\mathbf{y}) = (4\pi t)^{\frac{n}{2}} \exp\left(-\tfrac{1}{t} \arccos^2\left(\sum_{i=1}^{m} \sqrt{\mathbf{x}_i \mathbf{y}_i} \right) \right) \tag{2.49}$$

Here, n is a free hyperparameter, part of a positive exponent, and no longer bound to the dimension of the data; the constant m is the original dimension of the data vectors. Kernel K_t^{ID} arises as a special case of $K_{n,t}^{IDS}$, by setting n to $-m$.

2.5.2 Negative geodesic distance

In Zhang et al. [2005], a simple, hyperparameter-free geodesic kernel is proposed: the negative geodesic kernel NGD

$$K_{NGD}(\mathbf{x}, \mathbf{y}) = -2 \arccos \left(\sum_{i=1}^{n} \sqrt{\mathbf{x}_i \mathbf{y}_i} \right) \tag{2.50}$$

Notice that this kernel combines a *local*, Euclidean notion of similarity (the geometric mean, $\sqrt{\mathbf{x}_i \mathbf{y}_i}$) with a geodesic notion of similarity: the vector product expresses cosine similarity, while the inverse cosine expresses the measurement of distance along a curve. Two vectors that are completely similar will produce a kernel score of 0; two totally different vectors will produce $-\pi$. This is not strictly speaking a distance; however, the value $\mid K_{NGD}(\mathbf{x}, \mathbf{y}) \mid$ is a distance: the *positive geodesic distance*. We will refer to this distance with $K_{GD}(\mathbf{x}, \mathbf{y})$. Zhang et al. [2005] obtain state of the art results with the NGD kernel on a number of text classification tasks, outperforming the linear kernel applied to L2-normalized $tf.idf$ values. A *shifted* version of this kernel is

$$K_{NGD}^{+}(\mathbf{x}, \mathbf{y}) = -2 \arccos \left(\sum_{i=1}^{n} \sqrt{\mathbf{x}_i \mathbf{y}_i} \right) + \pi \tag{2.51}$$

taking values in the interval $[0, \pi]$.

Zhang et al. [2005] prove that the kernel K_{NGD} is conditionally positive definite, and that the shifted version is positive definite. The kernel K_{IDK}, on the other hand, was proved not to be positive definite by Martins et al. [2008]. Zhang et al. [2005] found no use in tweaking the C (cost) hyperparameter of the quadratic optimization problem solver.

2.6 Summary

In this chapter, we have presented an introduction to the basic concepts and techniques of machine learning that are relevant to this thesis. In addition, we presented a hyperparameter estimation procedure based on the cross-entropy method, and compared it in a number of experiments with a progressive sampling-based grid search algorithm. Subsequently, we discussed Euclidean distance-based vector space models for document classification, and the geodesic techniques underlying the multinomial document classification framework. A simple

transformation, L1-normalization, turns frequency-based bag of words representations of documents into objects that are embedded in a Riemannian manifold: a curved information space. This manifold is related through a simple pullback operation to the multinomial simplex, the parameter space of the multinomial distribution. On this manifold, distances between points are measured using geodesic metrics. Results from the field of text classification show that these geodesic measures are better approximations of interdocument distances, and therefore warrant the multinomial approach to document classification as a geometry-sensitive approach. In Chapter 3, we will evaluate this approach on two document classification tasks.

Chapter 3

Multinomial text classification

In this chapter, we describe two text classification experiments with geodesic kernels. In Section 3.1, we address the classification of sentences as either expressing an opinion (a subjective point of view) or a factual, objective statement. We obtain state of the art results using L1-normalized frequencies of character n-grams. In Section 3.2, we apply a similar approach to classification of entire blog posts as either positive or negative (the 2008 Blog Trec task). The latter task is known as polarity classification.

3.1 Subjectivity analysis

Given the huge growth of the blogosphere, and the increasing awareness by the industry of the merits of mining social media, sentiment analysis –the automated analysis of sentiments and opinions– has firmly established itself on the research agenda. Applications are diverse, ranging from analyzing customer feedback in blogs, web forums and e-mails to security investigations and personalized advertising. This implies there are many approaches to analyzing sentiment. One can focus on generating alerts for negative opinions given a certain product or person, on exploring the process of opinion-formation around a certain topic (identifying opinion leaders), or on modeling the flow of sentiment in a certain textual domain, for instance.

Global sentiment classification, the classification of entire documents as being (e.g.) positive, negative or neutral on average, can be viewed to a large extent as a document classification problem. While sentiment seems to be a too intricate notion to capture in terms of crude categories such as positive, neutral or negative, this is not really different from any classification task, where class labels usually only partially cover the semantics of objects to be classified. The global sentiment of an entire document is built up from several opinions, usually expressed on (or even below) the sentence-level, and it is not a priori clear how to exactly compute the aggregate sum of sentiments for the entire

document. Similarly, in document classification, the global topic of a text may be split up into several subtopics as well. Sentiment classification can be seen as a rough approximation of the sentiment in a document, ignoring the exact dynamics of sentiment flow, and its structural decomposition into linguistic units like sentences.

Subjectivity classification involves the discrimination between subjective and objective utterances, such as sentences, or even phrases. Subjective utterances reflect a private point of view, emotion or belief. Recognition of subjectivity is important from several points of view. Research in the field of sentiment mining (Pang and Lee [2004]) has shown that removing objective sentences from text prior to applying sentiment classification yields higher classification accuracy. Subjectivity classification is important for product review mining (Hu and Liu [2004], Kim and Hovy [2006]). Both summarization and information extraction (Seki et al. [2005], Stoyanov and Cardie [2006]; Riloff et al. [2005]) benefit from an adequate discrimination between subjective and objective content.

In this section, we present a shallow linguistic approach to subjectivity classification (Raaijmakers and Kraaij [2008]). Using Support Vector Machines equipped with geodesic kernels, we demonstrate that a data representation based on counting character n-grams is able to improve on results previously attained on the MPQA corpus using word-based n-grams and syntactic information. Specifically, we compare two types of string-based representations: key substring groups and character n-grams. Bias-variance decompositions of the errors made by the various classifiers show that word-spanning character n-grams substantially reduce the bias error of a classifier, and boost its accuracy.

We view subjectivity classification as a regular text classification problem. For a well-known annotated dataset, we present empirical evidence that shows how shallow representations consisting of character n-grams reduce the bias error of a classifier, and leads to higher accuracies than obtained with more elaborate linguistic features. We start out by discussing shallow, string-based representations of language, and subsequently present an array of experiments.

3.1.1 Shallow linguistic representations

The question how much high-level or *deep* linguistic structure (such as syntactic, or semantic information) is necessary for machine learning approaches to language analysis is part of an historical and ongoing debate. Already in 1957, Luhn proposed to represent text simply by a vector of weighted terms (Luhn [1957]). This eventually led to several document ranking models such as the probabilistic model of Robertson and Jones [1976] and the vector space model of Salton et al. [1975]. Document classification turned out to be feasible using only *shallow*, simple vector space representations of normalized or weighted frequencies of words, irrespective of word order or phrasal context.

Recent research in text analytics has provided further motivation that shal-

low linguistic representations are able to capture important linguistic aspects of utterances, while being far easier to compute and presupposing less linguistic theory. For instance, Giuliano et al. [2006] show that shallow linguistic features consisting of words, lemmas and orthographic equivalence classes (capitalized words, numerals, etc.) are quite useful for relation extraction in the biomedical domain, and outperform approaches that deploy syntactic and semantic information. Stamatatos [2006] uses character n-gram models to successfully predict authorship, using ensemble classifiers. Li and Roth [2001] observe that shallow parsers provide for better performance and robustness when confronted with new and low quality texts than their 'deep linguistic' counterparts. Kanaris and Stamatatos [2007], propose the use of character n-grams for webpage genre identification. Mcnamee et al. [2001] used word-spanning character n-grams in the HAIRCUT document retrieval system at TREC-9. The early work of Cavnar and Trenkle [1994] evaluates character n-grams for document classification. The LingPipe suite[1] uses character n-grams for sentiment classification.

3.1.2 Attenuation

String-based feature representations often suffer from a high level of specificity. Finding a suitable level of abstraction from raw feature values that still captures important distributional properties of the underlying data is an area of active research in the machine learning community. Eisner [1996] presents a form of feature value abstraction he calls *attenuation*: in order to deal with unknown words for dependency parsing, low-frequency unknown words are converted to high-level descriptors describing orthographical properties (such as capitalization, digits, etc.). Unknown name are described by `MORPH-CAP`, for instance; words ending in digits are described by `MORPH-NUM`. This leads to equivalence classes of tokens. In van den Bosch and Buchholz [2001], this technique is used for shallow parsing with memory-based learners.

The work of Macskassy et al. [2001] has applied binning (discretization) to numeric data, followed by the application of textual classification methods. For several numerical classification tasks from the UCI data repository, numerical values were discretized to symbolic values indication the bin into which a certain value falls. This creates a bag of word-like representation that can be directly used by regular text classification methods that operate on this type of data. Binning can be seen as a form of attenuation, grouping numerical features into equivalence classes (bins). The bias error of the resulting classifiers was shown by Macskassy et al. [2001] to be much lower than the bias error of classifiers operating on the original numerical data. In the biomedical domain, the work of Settles [2005] proposes the use of regular expression-style ('regex') features for protein and gene tagging in biomedical texts using Maximum Entropy models. These regex features abstract from specific orthographic properties of strings by

[1] Available from http://www.alias-i.com/lingpipe/

grouping characters into character equivalence classes, such as `[A-Z]` (capitalized characters) or `[0-9]` (digits). Leslie and Kuang [2004] use specific kernels (such as wildcard kernels) that allow for inexact string matching, thus performing a similar kind of abstraction.

Zhang and Lee [2006] present a *key substring group* representation for document classification purposes that significantly outperforms approaches based on deep linguistic analysis. Given a text corpus, a suffix trie is formed that compresses the entire corpus in a tree labelled with set-valued nodes. These nodes are containers for substrings that have exactly the same distribution in the given corpus: a key substring group is a set of substrings that share the same path label in the trie that stores them. By definition of the trie data structure, all strings in a key substring group have exactly the same frequency. Since these substrings are equivalent from a distributional point of view, they can be safely replaced by a single arbitrary symbol, which constitutes another form of attenuation based on distributional string equivalence. The author reports significant improvement of state-of-the-art results, including a 20% improvement over deep linguistic (morphological and syntactical) analysis of Greek authorship classification, and the best result ever reported on the ten largest categories of the Reuters-21578 dataset. For these tasks, character n-gram approaches appeared to fare quite well, but not as good as key substring groups.

3.1.3 Character n-grams

In statistical language modeling (Rosenfeld [2000], Jelinek [1997]), language models assign probabilities to sequences of tokens $t_1 \dots t_N$ according to a product of local probabilities:

$$P(t_1 \dots t_N) = \prod_{i=1}^{N} P(t_i \mid t_1 \dots t_{i-1}) \tag{3.1}$$

which, in the case of an n-gram representation ($n = 1...N$) becomes

$$P(t_1 \dots t_N) = \prod_{i=1}^{N} P(t_i \mid t_{i-n+1} \dots t_{i-1}) \tag{3.2}$$

Maximum likelihood (ML) estimates for n-grams are determined by relative frequencies of cooccurrence:

$$r(t_i \mid t_{i-n+1} \dots t_{i-1}) = \frac{\mid t_{i-n+1} \dots t_i \mid}{\mid t_{i-n+1} \dots t_{i-1} \mid} \tag{3.3}$$

These ML estimates can be made much more accurate when the amount of data increases. In general, n-gram models converge to the ML estimate with enough

data, under smoothing schemes that usually reserve a small probability mass to assign non-zero probabilities to unseen events. Clearly, the more data these models see, the less probability mass is consumed by unseen events. Generating more data by using longer n-grams (5-grams, 6-grams, etc.) is not a realistic option, as this will lead to sparse event spaces: the chances of observing a certain n-gram decrease with increasing values of n. An easy way to generate more data is to exploit the subword level and use character n-grams. For instance, for the following sentence

$$\textit{This car really rocks.} \tag{3.4}$$

subword character bigrams and trigrams are

- *th*, *hi*, *is*, *ca*, *ar*, *re*, *ea*, *al*, *ll*, *ly* , *ro*, *oc*, *ck*, *ks*, *thi*, *his*, *car*, *rea*, *eal*, *all*, *lly*, *roc*, *ock*, *cks*.

Any n-character word produces $n-1$ character bigrams and $n-2$ character trigrams. This means that for any document D containing w words with average length l, the rough expansion factor is $w \cdot (l-1) + w \cdot (l-2) = 2wl - 3w$ for character bigrams and trigrams.

In discriminative (non-generative) models of machine learning, n-gram information typically is used in the form of a *bag of words* (bow), or, better put, a bag of n-grams. The bow model assumes a feature space consisting of a set of frequency counts, and treats every feature in this feature space as an element of a multiset. The bag $\{a, b, a, b, b\}$ can be interpreted as a frequency table $a : 2, b : 3$, and is open to term weighting methods, such as $tf.idf$ (see e.g. Joachims [1998]).

Intuitively, more items in the bag make the bag more informative, as it makes for a more descriptive event space and presumably a better model. So, more data might be benificiary for discriminative models as well. Yet, this is an intricate issue, as words that are not strongly predicative of a certain class may lead to conflation problems with other classes. This is the reason why in many applications (e.g. Ng et al. [2006] and Raaijmakers [2007]), term selection is applied in order to create a strongly class-predictive index vocabulary. Further, the number of model parameters should be optimized according to the trade-off between bias and variance errors: a low number of model parameters may increase the bias error of the model (the systematic errors it makes), and a high number contributes to high variance error (the data-dependent errors) and leads to an overfitted model.

The bag of word approach ignores sequentiality altogether[2]. Nonetheless it is clear that there is useful information in the sequential structure of documents. For sentiment mining, specific combinations of terms appear particularly

[2]External information, such as a language model, allows to reconstruct some of the sequential structure underlying in a bag of words.

informative, most notably valence shifting combinations such as 'not good' (Andreevskaia et al. [2007]). A simple (but limited) way of restoring sequentiality for a character-based bow approach is to employ n-grams on the subword level that span word boundaries and therefore reach the superword level. For instance, a bigram and trigram representation for sentence (3.4) that span word boundaries produces

- *th*, *hi*, *is*, *s#*, *#c*, *ca*, *ar*, *r#*, *#r*, *re*, *ea*, *al*, *ll*, *ly*, *y#*, *#r*, *ro*, *oc*, *ck*, *ks*, *thi*, *his*, *is#*, *s#c*, *#ca*, *car*, *ar#*, *r#r*, *#re*, *rea*, *eal*, *all*, *lly*, *ly#*, *y#r*, *#ro*, *roc*, *ock*, *cks*

with # a whitespace indicator. These n-grams capture transitions between consecutive words, and thus encode phrasal effects on character level. Notice that the amount of string data increases significantly (39 vs. 24 character n-grams). For w words, the expansion factor for bigram and trigram superword character n-grams is $2(w-1)+3(w-1) = 5w-5$ extra strings. We shall refer to these word-spanning n-grams as *superword character n-grams* (supergrams, for short) and to word-internal character n-grams as *subword character n-grams* (or subgrams). As character n-grams do not encode positional information, a limited form of attenuation arises naturally from string overlap, where similar n-grams coming from different words are considered to be the same, forming equivalence classes not unlike regular attenuation.

In our recent work (Raaijmakers and Kraaij [2008]; Raaijmakers and Wilson [2008]; Raaijmakers et al. [2008]) we have found ample evidence for the informativity of character n-grams for sentiment analysis. In Raaijmakers et al. [2008] we demonstrated for a large array of experiments that character n-grams are the most informative source of information compared to phonemes, prosody and word n-grams. The role of prosody as compared to lexical features for sentiment classification was also investigated in Truong and Raaijmakers [2008].

3.1.4 Data and experiments

Our data consists of the MPQA corpus (Wiebe et al. [2005]), a corpus composed of 535 news articles from 187 foreign and US sources, manually annotated for sentiment according to the Multi Perspective Question Answering annotation scheme (Wiebe et al. [2005]). This annotation scheme is based on the notion of *private state* (Quirk et al. [1985]). A private state models the state of an experiencer displaying an attitude towards a certain target. Annotations have been made at the phrasal level; higher-level annotations at the sentence level are derived from these lower-level annotations. We extracted 9,266 sentences from the lower-level annotations of this dataset. From the specifications of Riloff et al. [2006], a sentence is subjective if it contains a subjective expression of medium or higher intensity. We counted 5,010 subjective sentences and 4,256 non-subjective sentences (54% of the data is subjective). According to this

definition, sentences are labelled as subjective on the basis of minimally one subjective expression. Sample subjective sentences from the MPQA corpus are:

- *Under these circumstances, Taiwan and Japan both must recognize clearly the important role of the navy.*
- *So when the White House says it is going to treat the Taleban in accordance with the Geneva Convention, it is pure rhetoric, nothing other than a public relations exercise and will be meaningless in practice.*

Some objective sentences are:

- *By the time of Mao Tse-tung's Moscow speech the Chinese were actively searching for a site for their rocket range.*
- *McKerrow's estimates are based on household counts coupled with mathematical models used by the WHO.*

In a number of experiments, we set out to answer the following question: are shallow linguistic representations adequate for subjectivity classification at the sentence level? Sentences being relatively short, using shallow representations based on words (such as unigrams and bigrams) might perform not as good as representations based on the much richer information space of substrings. The work of Riloff et al. [2006], discussed in Section 3.1.7, is revelant to our work, as it addresses subjectivity classification of the MPQA corpus. These authors kindly made available their exact data fragment from the MPQA corpus, from which we derived a number of substring-based representations:

- Subword character n-grams (bi-, tri- and quadrigrams)
- Superword character n-grams (bi-, tri- and quadrigrams)
- Key substring groups
- A mixture of key substring groups and superword character n-grams (bi-, tri- and quadrigrams)

The word-internal character n-grams were used as a baseline. The key substring group features were generated with standard parameter settings of the software made available by Zhang and Lee [2006][3]. Due to an inherent memory restriction of this software, we had to (uniformly) limit training set size over all runs and data representations to 5,000 datapoints, which amounts to 81% of the original training data. In the experiments reported in this section, we use the negative geodesic kernel K_{NGD} applied to normalized 'count once' frequencies (every feature out of a total of n obtains a probability $1/n$), following the

[3]Software available from
`http://www.dcs.bbk.ac.uk/~dell/publications/dellzhang_kdd2006_supplement.html`.

work of Yang et al. [2007] on bags of visual words, where it was demonstrated empirically that for large vocabularies, binary frequencies outperform regular frequencies. In our case, the character n-gram representation leads to a large vocabulary size. This type of representation effectively is a maximum entropy representation, as the probability distribution is flat. In Chapter 6 and Chapter 7 we will argue that this type of representation may be better learnable with non-geodesic kernels, but that is not our main concern in this section.

3.1.5 Results

Results in Table 3.1 show first of all that superword character n-grams perform the best among the other representations. Even on the basis of 81% of the training data, this representation also leads to improvement of results reported on this data by Riloff et al. [2006] using unigram, bigram and extraction pattern features.

	SUB	SUPER	KSG	KSG + SUPER
Acc	74.57	**82.5**	77.9	81.9
Rec	77.6	**84.9**	80.7	84.4
Prec	75.9	**83.2**	78.9	82.5
F_1	76.7	**84**	79.8	83.5

Table 3.1: Average accuracy, recall, precision and F_1 (three-fold cross-validation) for subgrams, supergrams, key substring group features and a combination of key substring group features and supergrams (best results in bold).

We graphically compare the four different data representations across the three cross-validation folds using cost curves (Drummond and Holte [2006]; see Appendix B). The curves in Figure 3.1 confirm the superior performance of superword character n-grams (lower lines indicate better performance; the operating point 0.5 indicates equal costs for both classes (subjective vs. objective), and the values on the y-axis reduce to error rate. The combination of key substring group features and superword character n-grams performs worse than superword character n-grams alone. The baseline consisting of word-internal character n-grams performs worst.

3.1.6 Bias and variance decomposition of classification error

As has been noted by Webb [2000], the implementation of bias and variance error presented by Kohavi and Wolpert [1996] suffers from the fact that training subsets become very small for medium to average size data sets, and the

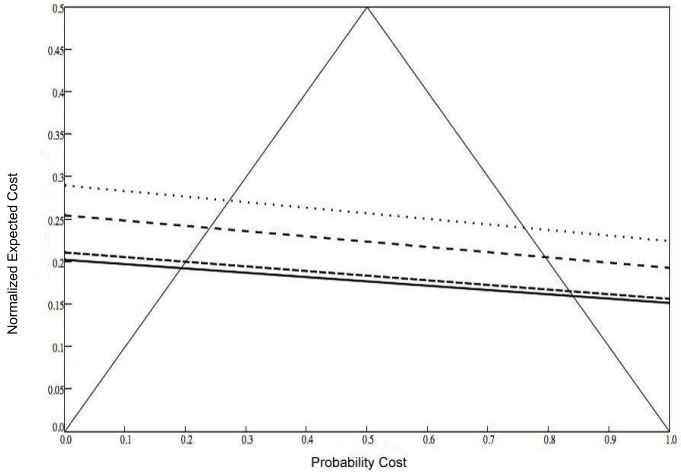


Figure 3.1: Combined cost curves for 3 folds, comparing superword character n-grams (solid line), subword character n-grams (dotted), KSG (long dashed), and KSG+superword character n-grams (short dashed).

estimates for bias and variance error consequently become unreliable. We took to heart the recommendations of Webb [2000], and applied 10 runs of 3-fold cross-validation, in order to get better estimates of bias and variance. From Table 3.2, we see that supergrams yield lower bias error than subgrams and key substring groups. The combination of key substring groups and superword character n-grams has a slightly lower bias error than superword character n-grams alone, but it this difference is probably not significant. As noted, overall performance of this combination is lower than the performance of superword character n-grams alone. From the variance error decomposition, it appears that supergrams reduce the systematic error of a classifier more than its data-dependent error. The variance error produced by supergrams is lower than subgrams, but slightly higher than for KSG or the combination of KSG and supergrams. All in all, the combined advantage of lower bias error and comparable variance error of supergrams compared to subgrams and KSG-based representations is quite clear.

3.1.7 Related work

A significant body of work has addressed subjectivity classification, ranging from phrase level, to sentences and the document level. In this overview, we

	SUB	SUPER	KSG	KSG + SUPER
Bias	39.8	35.9	37.9	35.8
Variance	27.9	27.7	27.5	27.6

Table 3.2: Bias and variance decomposition of the classification error, using subgrams, supergrams, key substring group features and the combination of key substring group features and supergrams

limit ourselves to work addressing the sentence and phrase level.

Kim and Hovy [2004] investigate a number of metrics that combine lexical sentiment polarities at the word level into an aggregate polarity score for entire sentences. Their best approach is a polarity product method that effectively cancels out negative-negative combinations in a certain region (negative times negative is positive). They conclude that the presence of negative words is more important than their exact polarity strengths. Their method is basically a lexical method. The interaction between word senses (meaning) and subjectivity is studied by Wiebe and Mihalcea [2006], arguing for a direct relation between lexical subjectivity and the use of subjectivity information for word sense disambiguation. Similarly, Yu and Hatzivassiloglou [2003] take a lexical approach to subjectivity classification of both sentences and entire documents. The authors detect subjective sentences in TREC data using word unigrams, bigrams, parts of speech and frequency counts of subjective words, obtaining 91% F-measure scores with a Naive Bayes classifier at the sentence level, and up to 97% F-measure scores at the document level. Their work illustrates the difficulty of identifying subjectivity at the sentence level.

The focus of Wilson et al. [2006] is on intensity classification: classifying the strength of opinion using a five-point scale (neutral, low, medium, high, extreme), after first detecting opiniated texts. This work addresses also clauses below the sentence level. Likewise, the work of Wilson et al. [2005] is concerned with phrase-level subjectivity classification and sentiment classification. In addition to a bag of three tokens (previous, current and next word, on the basis of a shifting window), contextual features addressing parts of speech and syntactic dependency are used. After the discrimination between subjective and neutral phrases, polarity classification is carried out. Breck et al. [2007] describe phrase-level recognition of subjectivity (below sentence-level) in the MPQA corpus, using conditional random fields that treat the task as a tagging problem.

Wiebe et al. [1999] extract subjective nouns from unannotated text on the basis of manually created lists of seed words, and use these nouns together with a wide range of previously identified interesting features for sentence-based subjectivity classification. Most notably, these features include strongly class-predictive stems: words in their base form that are strongly associated with

either the subjective or objective class. Riloff et al. [2003] also investigate subjectivity classification at the sentence level. They apply rule-based classifiers based on lexical lookup and simple decision rules to bootstrap labelled training data from unlabelled data, on which extraction pattern learning algorithms are trained. These algorithms feed back their output to a training set, and the process is reiterated. Recall and precision scores on a 2,197 sentences test set range from 32.9–40.1 and 90.2–91.3 respectively.

Riloff et al. [2005] use rule-based classifiers to bootstrap training material for a Naive Bayes classifier for sentence level subjectivity classification. Their system is evaluated in the context of MUC-4 terrorism data, the Naive Bayes classifier producing information extracts on the basis of subjectivity. Using additional filtering contraints, they obtained higher precision and only slightly lower recall compared to previous attempts, thus illustrating the usefulness of sentence-level subjectivity classification.

The issues of feature construction and feature selection are addressed by Riloff et al. [2006] for sentence-level subjectivity classification. Observing subsumption relations between many feature types (e.g. bigrams are subsumed by the unigrams they are composed of), these authors propose a subsumption hierarchy for features, which subsequently can be used for feature selection purposes: complex features can have different (sometimes better) predictive properties (higher information gain) than their simple parts. They reported a best accuracy of 74.9% (three-fold cross-validation) using either a combination of word unigrams and word bigrams, or word unigrams and bigrams together with syntactic extraction patterns. Li et al. [2007] report an F-score of 77.9 on a single training-test split of the MPQA corpus, using Support Vector Machines and raw token unigrams, lemma unigrams and part of speech information.

3.1.8 Conclusions

In this section we have evaluated shallow, character-based representations of subjectivity data. Character-based representations are easy to compute, and presuppose a minimum amount of linguistic theory: basically, tokenization is the only prerequisite. We compared character n-gram representations with key substring group representations, and provided empirical evidence for the superior performance of superword character n-grams: character n-grams that surpass word boundaries. We found that these n-grams substantially reduce the bias error of a classifier, and also outperform the combination of shallow and deep linguistic features used by Riloff et al. [2006]. A logical continuation of this work is to apply the technique to local sentiment on the phrase level and to investigate the combination of deep and shallow information in one classifier. In Chapter 4, we will demonstrate that carefully balancing (interpolating) heterogeneous information in a multinomial classifier applied to sentiment classication is beneficiary.

3.2 Large-scale polarity classification

In this section we describe the TNO submission (Raaijmakers and Kraaij [2009]) to the Blog TREC 2008 task of large-scale polarity classification. We submitted 5 runs provided by NIST, to which we applied information diffusion kernels operating on character n-gram representations.

3.2.1 Introduction

The polarity task of Blog TREC 2008 consists of retrieving and ranking for each of a total of 150 topics (queries) the positive and negative opiniated documents in the test collection. TREC has made available 5 topic-relevance baseline runs, to which polarity classification or opinion finding techniques can be applied. This allows participants to focus on one aspect of the processing chain. In this contribution, we describe the result of applying the TNO polarity classification approach to these 5 baselines. We discuss the results of our submissions in Section 3.2.5 and present conclusions and lessons learned in Section 3.2.6. In the next three sections, we describe the data, our feature representation, and the general outline of our setup.

3.2.2 Data and pre-processing

The TREC Blog06 collection, a 148 Gigabytes sample of the blogosphere, is the result of an eleven-week period crawl (December 2005-February 2006). Due to the automated crawling process, the dataset contains not only legitimate blog postings, but also spam, javascript, homepages and RSS feed material. The data itself consists of raw HTML, with a total of over 3.2 million documents. In order to train a classifier on these class-labeled web pages, these documents have to be cleaned up and converted to plain text, which is by far not a trivial task. Our HTML to text conversion strategy consists of a dedicated DOM-parser effectively stripping the larger part of HTML tags and javascript code. We combined this parser with the `html2text` Python script[4] in sequence: following our dedicated parser, we applied `html2text.py`. While this produced reasonably clean text, we found that in a lot of cases the output data still contained tags and programming constructs.

3.2.3 Character n-gram representations

As for the subjectivity analysis presented in Section 3.1, we opted for a character n-gram approach to the polarity classification task. For every training document, we generated superword character n-grams from 2 up to 6 characters. In

[4] Available from http://www.aaronsw.com/2002/html2text/

the experiments reported in this section, we used the hyperparameter-free negative geodesic kernel K_{NGD}. Expanding the TREC data to character n-grams leads to a huge expansion of data. Due to memory constraints of our systems, we took a random portion of training data of only 16% (amounting already to over 250 megabytes of training data).

3.2.4 Thresholding decision values

Support Vector Machines output decision values that can be discretized to binary classes (a negative value produces a negative class label, and a positive value a positive class label), or posterior class probabilities (e.g. Platt [2000], and Chapter 8). We performed no such discretization, but used the raw decision values for ranking the various positive and negative cases. We devised a simple threshold estimator that, on the basis of class distribution priors in the training data, determines the optimal threshold above which decision values should produce positive classes. Algorithm C.3 performs a pseudo-exhaustive one-parameter sweep, fixing a decision value threshold that optimally approximates the *a priori* class distributions in the training data. The d_{min}, d_{max} demarcate the range of the decision values in the training data. The ρ factor is a resolution factor that serves to round off obtained class distribution scores. We used this threshold to assign classified documents to the positive and negative classes, prior to ranking their respective decision values. The value of σ was set to 0.05.

3.2.5 Results

In Figure 3.2 and Figure 3.3, the results for positive and negative queries are displayed, by plotting the difference of the produced MAP and R-PREC values[5] and the reference values. As can be seen, the runs for the positive queries produce well above median scores for both MAP and R-PREC. Averaged over the 5 baseline runs, for the positive queries, a portion of 62.3% is equal to or above the reference median average precision. For the R-PREC scores for positive queries this portion is on average 70.5%. The R-PREC scores produced by the 5 positive baseline runs were all significantly[6] better than the median R-PREC reference scores. The average difference between produced R-PREC and reference R-PREC was +12.1%. For the negative queries, on average, 24.7% of all MAP scores produced were equal to or above the reference MAP values. For R-PREC, a much higher proportion of on average 57.8% scores was equal to or above median reference R-PREC. The averaged difference over all 5 runs

[5] Mean Average Precision and Precision at R (with R the number of relevant documents); see Appendix B.

[6] All significance results were computed with the non-parametric Wilcoxon signed rank test, with $p < .5$.

for R-PREC compared with reference R-PREC was -1.7%. In 4 out of 5 runs, this difference was significant, its average amounting to a rather small -1.7%.

The percentages of deviations are listed in Table 3.3, as well as the results of the Wilcoxon signed rank applied to the R-PREC results.

Task	% MAP	% R-PREC	MAP	R-PREC	W
Positive queries					
base1	+66.9	+75.7	26.2	21.3	+14.7
base2	+53.4	+60.8	19.4	15.4	+7.9
base3	+64.2	+73	24.1	19.4	+12.5
base4	+64.2	+71.6	23.8	19	+12.3
base5	+62.8	+71.6	24.8	19.7	+13.2
Average	+62.3	+70.5	16.2	11.5	+12.1
Negative queries					
base1	+22.7	+61	8.7	4.5	-1.3
base2	+14.9	+49.7	6.7	3.5	-3.3
base3	+27.7	+58.9	7.7	4	-2.3
base4	+31.2	+59.6	8.3	4.5	-1.6
base5	+27	+59.6	8.4	4	=
Average	+24.7	+57.8	10	6.7	-1.7

Table 3.3: Percentage of queries with MAP scores above/below (+/-) median average precision; percentage of queries with R-PREC scores above/below median R-PREC; average MAP and average R-PREC scores for the 5 polarity baselines; Wilcoxon significance of the difference of the obtained score with the reference score ($p < .5$), as well as the difference of the average obtained score with the average reference score.

3.2.6 Conclusions

In this section, we presented the TNO approach to polarity classification and ranking of the Blog TREC 2008 data. For 5 baseline runs, we applied information diffusion kernels to character n-gram representations. We trained our system on a relatively small portion of 16% of the total available training data. Results show that our system performs well above median for positive queries. For negative queries, results are in 4 out of 5 runs below median, albeit with a small (but significant) percentage. While it is definitely necessary to invest more time in thorough data cleaning prior to classifier training and testing, these results underline the efficacy and accuracy of multinomial approaches to document classification.

3.3 Summary

In this chapter, we have demonstrated the usefulness of multinomial classifiers. Applied to the problem of sentiment analysis, we obtained state of the art performance for subjectivity classification on sentence level. Character n-gram expansion was applied to generate a sufficient amount of data for the multinomial classifiers. For the Blog Trec task, we obtained well above median performance for positive blog posts, and slightly below median performance for negative blog posts. Given the fact that we used a mere 16% of the training data, and that we only partially succeeded in cleaning up the training and test data, this result demonstrates the potential of these kernels when applied to only partially cleaned up text. Apart from the research reported in this chapter, we applied multinomial classifiers to the problem of re-ranking answers to *why*-questions (see Verberne et al. [2009]). In Chapter 4 and Chapter 5, we will propose extensions to the standard multinomial framework in order to be able to deal with two phenomena: heterogeneous information, and sequential and limited context problems.

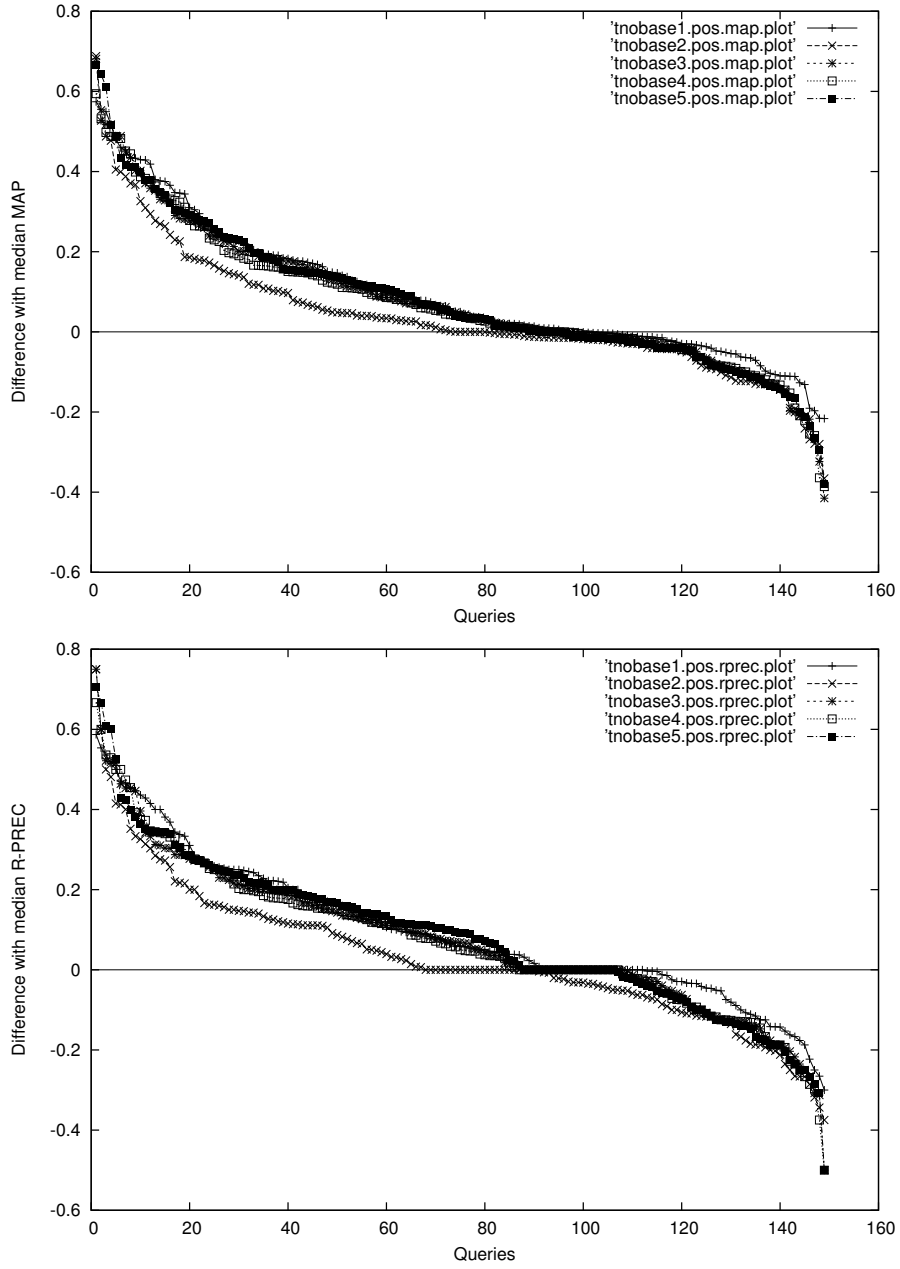


Figure 3.2: Difference of TNO produced MAP and R-PREC values with the TREC reference values for positive queries (queries sorted by descending performance), for the five submitted runs.

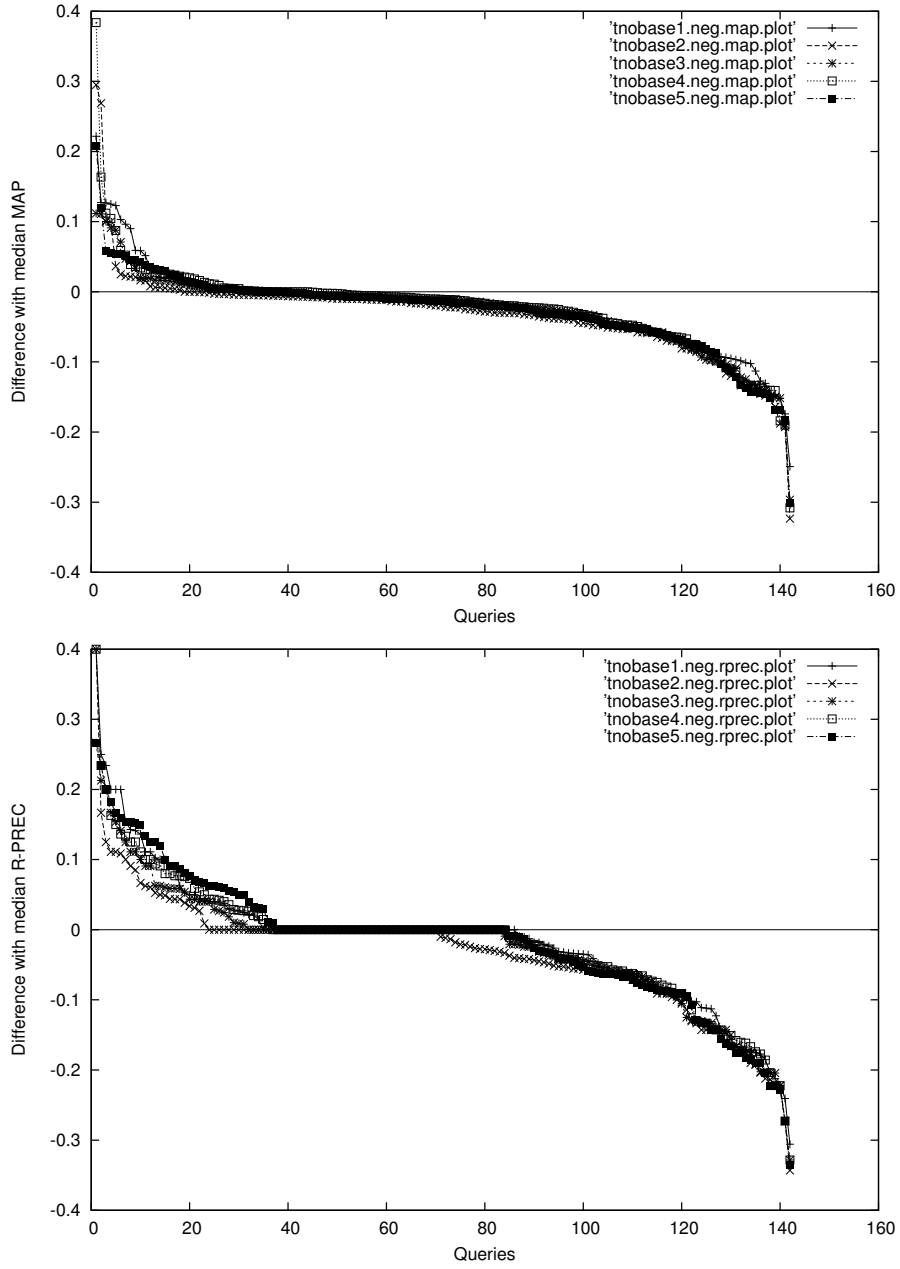


Figure 3.3: Difference of TNO produced MAP and R-PREC values with the TREC reference values for negative queries (queries sorted by descending performance).

Chapter 4

Hyperkernels

4.1 Introduction

Up to now, we have seen the application of geodesic kernels to L1-normalized frequencies of string data. Both for the tasks of subjectivity classification and polarity classification, we have applied the negative geodesic kernel to heterogeneous data: bags of strings drawn from different multinomial distributions, mixing uni-, bi- and higher-order n-grams without keeping them apart. While e.g. word unigrams and bigrams are related through subsumption, it seems counterintuitive to normalize the frequencies of higher n-grams with the frequencies of lower-order n-grams. Revisiting our previous example

$$\begin{array}{l} th, hi, is, ca, ar, re, ea, al, \\ ll, ly, ro, oc, ck, ks, thi, his, \\ car, rea, eal, all, lly, roc, ock, cks. \end{array} \tag{4.1}$$

we get a normalized frequency of $\frac{1}{24}$ for every string (all strings (24) are hapaxes). However, in this example there are only 10 trigrams, opposed to 14 bigrams, which shows that the normalized trigram frequency is dominated by the bigram frequencies.

It has been demonstrated in a number of publications (Section 4.3.4) that n-gram features bear quite different information. In this chapter (based on Raaijmakers [2007]), we therefore raise the question whether it is beneficiary to discriminate between different sources of information in the classification process, specifically when using L1-normalized data and geodesic classifiers. We will start with a formal analysis of weighted information as a form of kernel interpolation, after which we will present empirical evidence for the fruitfulness of this approach.

4.2 Kernel interpolation

In the language modeling field, mixing different types of information is usually done by interpolation (e.g. Kraaij [2004]) and mixtures of Gaussians (e.g. Dasgupta [1999]) in generative models. While our framework of kernel-based classification is *discriminative* as opposed to *generative* (Jebara [2001]), it turns out there is a simple method for implementing similar ideas: kernel interpolation. Kernel interpolation is the process of combining two or more kernels in a weighted manner. Every kernel receives its own weight that expresses the importance of the contribution of this kernel in the classification process. The idea for our problem is to have two similar kernels both addressing different types of information: one for word unigrams, and one for word bigrams.

While there are many ways to combine heterogeneous information in one kernel, the following composed kernel combines information through a weighted sum, factoring out the contributions of two separate Bhattacharyya subkernels. The weights λ_1 and λ_2 are interpolation weights that express the relative importance of the two information sources.

$$K^{IDS}_{n,t,\lambda_1,\lambda_2}(\mathbf{x},\mathbf{y};i,j) =$$

$$(4\pi t)^{\frac{n}{2}} \exp\left(-\tfrac{1}{t} \arccos^2\left(\frac{\lambda_1\left[\sum_{k=1}^{i} \sqrt{\mathbf{x}_i\mathbf{y}_i} \right] + \lambda_2\left[\sum_{i+1}^{j} \sqrt{\mathbf{x}_i\mathbf{y}_i} \right]}{2} \right) \right) \tag{4.2}$$

The two subscript parameters i and j indicate the highest index positions of the word unigrams and bigrams in the vocabulary. This information is used by the kernel to factorize the unigram and bigram parts: any feature with an index higher than the maximum unigram index will be considered a bigram feature.

As we will see in Section 4.2.1 and Section 4.2.2, this method has at least two mathematical interpretations.

4.2.1 A pullback metric

Weighting the Bhattacharyya subkernels corresponds to computing a weighted inner product, which effectively weighs an entire feature subspace with uniform weights:

$$\lambda(\sum_{i=1}^{N} \sqrt{\mathbf{x}_i\mathbf{y}_i}) = \sum_{i=1}^{N} \sqrt{(\lambda\mathbf{x}_i)(\lambda\mathbf{y}_i)} \tag{4.3}$$

From this observation, we derive the following

4.2.1. Proposition. *Kernel interpolation corresponds to a pullback metric on the Fisher information.*

Proof The proof of the proposition is direct, by the following:

$$G_\lambda^c(\theta) = \left(\frac{\theta_i \lambda}{\sum_{i=1}^{n+1} \theta_i \lambda}, \ldots, \frac{\theta_{n+1} \lambda}{\sum_{i=1}^{n+1} \theta_i \lambda} \right) \tag{4.4}$$

■

4.2.2 Submanifold regularization

An alternative, formal interpretation of kernel interpolation is the following. A learning algorithm is a map that selects from a hypothesis space of functions $H : X \times X \mapsto \mathbb{R}$ a function $f^* : x \mapsto y$ such that $f(x) \approx y$ in a predictive way, with y the true class of x.

Assume a set of l labeled examples $(\mathbf{x}^i, y_i)$, with $\mathbf{x}^i$ a feature vector and $y_i \in \{+1, -1\}$ its class[1]. Mercer kernels $K : X \times X \mapsto \mathbb{R}$ are associated with a set of functions H_K called the Reproducing Kernel Hilbert Space (RKHS). These functions $f : X \mapsto \mathbb{R}$ have a norm $\| \cdot \|_K$, and penalizing this norm in the following formula amounts to smoothing solutions for f (Belkin et al. [2006]):

$$f^* = \arg \min_{f \in H_K} \frac{1}{l} \sum_{i=1}^{l} \Lambda(\mathbf{x}^i, y_i, f) + \gamma \parallel f \parallel_K^2 \tag{4.5}$$

with Λ a loss function, like the *hinge loss*

$$\mid 1 - y_i f(\mathbf{x}^i) \mid_+ \tag{4.6}$$

with

$$\mid x \mid_+ = max(x, 0) \tag{4.7}$$

This technique, known as Tikhonov Regularization (e.g. Schölkopf and Smola [2002]), is an extension of Empirical Risk Minimization (ERM; e.g. Vapnik [1999]), which leaves out the penalty term:

$$f^* = \arg \min_{f \in H_K} \frac{1}{l} \sum_{i=1}^{l} \Lambda(\mathbf{x}^i, y_i, f) \tag{4.8}$$

ERM is known to be ill-posed for unfortunate choices of H: f should not be easily perturbed by variation in the training data, it should be stable and continuously dependent on the data. Usually, however, H is unconstrained,

[1] As before, we use superscripts ($\mathbf{x}^i$ being the i-th vector $\mathbf{x}$) to avoid confusion with our previous subscript notation that refers to components of a vector.

and this condition is not trivially met. Tikhonov regularization (also known as manifold regularization) serves to constrain H with a penalty term.

We now show that the technique of kernel interpolation corresponds to a form of Tikhonov manifold regularization. As a prerequisite, note the following about a norm: a nonnegative function $\| \cdot \|$ is a norm iff $\forall f, g$ in a vector space N and $\alpha \in \mathbb{R}$

1. $\| f \| \geq 0$ *and* $\| f \| = 0$ iff $f = 0$
2. $\| f + g \| \leq \| f \| + \| g \|$
3. $\| \alpha \cdot f \| = | \alpha | \cdot \| f \|$

We now prove the following:

4.2.2. Proposition. *Regularizing a function that interpolates two kernels corresponds to a form of manifold regularization addressing two submanifolds.*

Proof Let $\mathcal{F}^+ \subset POW(\mathcal{F})$ be a set of feature subspaces of the feature space $\mathcal{F}$. Assume our classifier is a composite function $f = \frac{1}{2}[\lambda_1 f_{1|k} + \lambda_2 f_{2|l}]$, with $f_{i|j}$ a classifier restricted to feature subspace $j \in \mathcal{F}^+$. So, the f_1 and f_2 both address different slices of the feature space, which implies they address different manifolds. Notice that λ_1 and λ_2 are always positive, drawn from $\mathbb{R}^+$, hence $| \lambda | = \lambda$. We then have

$$
\begin{aligned}
f^* = \quad & \arg \min_{f \in H_K} \frac{1}{l} \sum_{i=1}^{l} \Lambda(\mathbf{x}^i, y_i, f) + \gamma \| f \|_K^2 = \\
& \arg \min_{f \in H_K} \frac{1}{l} \sum_{i=1}^{l} \Lambda(\mathbf{x}^i, y_i, f) + \gamma \| \frac{1}{2}[\lambda_1 f_{1|k} + \lambda_2 f_{2|l}] \|_K^2 = \\
& \arg \min_{f \in H_K} \frac{1}{l} \sum_{i=1}^{l} \Lambda(\mathbf{x}^i, y_i, f) + \gamma \| \frac{\lambda_1}{2} f_{1|k} + \frac{\lambda_2}{2} f_{2|l} \|_K^2 = \\
& \arg \min_{f \in H_K} \frac{1}{l} \sum_{i=1}^{l} \Lambda(\mathbf{x}^i, y_i, f) + \gamma \| \frac{\lambda_1}{2} f_{1|k} \|_K^2 + \gamma \| \frac{\lambda_2}{2} f_{2|l} \|_K^2 = \\
& \arg \min_{f \in H_K} \frac{1}{l} \sum_{i=1}^{l} \Lambda(\mathbf{x}^i, y_i, f) + \gamma | \frac{\lambda_1}{2} | \| f_{1|k} \|_K^2 + \gamma | \frac{\lambda_2}{2} | \| f_{2|l}] \|_K^2 = \\
& \arg \min_{f \in H_K} \frac{1}{l} \sum_{i=1}^{l} \Lambda(\mathbf{x}^i, y_i, f) + \frac{\gamma \lambda_1}{2} \| f_{1|k} \|_K^2 + \frac{\gamma \lambda_2}{2} \| f_{2|l}] \|_K^2
\end{aligned}
\tag{4.9}
$$

From this it follows that the penalty terms for the submanifolds directly depend on the choice for the interpolation weights (λ_1, λ_2), or, put differently, that optimizing the interpolation weights of the two submanifolds determines the solution of the regularization of f. ■

4.3 N-gram interpolation for global sentiment classification

The research question addressed in this section is the following. A substantial body of research has addressed the question which types of linguistic information are useful for sentiment classification, especially investigating the role of word unigrams and bigrams. We will take up this issue, and investigate the division of labor between word unigrams and bigrams for binary sentiment classification: classifying a document as either positive or negative. We start by presenting the ingredients for our experiments: the data, and the classifier apparatus. We then describe our experiments, and draw conclusions.

4.3.1 Data

We will focus on predicting the global sentiment for the annotated movie review dataset of Pang et al. [2002], commonly referred to as the *polarity dataset*. This dataset, consisting of 1000 positive and 1000 negative movie reviews, has become a *de facto* benchmark dataset for global binary sentiment classification. In Table 4.1, two sample snippets from a positive and negative review are listed.

Positive
Kidman knows a star-making role when she sees it and she plays the conscience-deprived Suzanne as the ultimate career driven, post-feminist uber-temptress from hell. She is ably abetted by wonderfully demented comic dialogue, which Kidman delivers in a unselfconsciously funny way.
Negative
Tons of illogical moments about; more than I really have the inclination to list. And I don't mean illogical in the summer popcorn movie sense. That type of illogical can be fun. This film insults your intelligence more times than I care to remember. The last 20 minutes especially degenerates into such lunacy that you'll be laughing more than you'll be screaming.

Table 4.1: Sample snippets of positive and negative movie reviews.

4.3.2 Combining linguistic information for sentiment mining

Many types of information have been identified as relevant for sentiment classification during the last years, e.g. sentiment-bearing word unigrams, word n-grams, valency information of adjectives, valency-shifting properties of adverbs, and dependency relations. In Section 4.3.4, we will discuss the relevant literature. In the work reported here, we concentrate on using word unigram and bigram information only, represented by frequency counts. These information sources are the natural ingredients for information diffusion kernels, once properly represented as normalized frequencies. Radial basis function (RBF) kernels, on the other hand, are frequently applied to either index vectors (where every feature flags the presence of a certain unigram or bigram), raw frequencies, or term weighting representations such as *tf.idf* (see e.g. Joachims [1998]).

Normalization of word unigram and bigram frequencies will produce two separate multinomial probability distributions that need to be reconciled during the classification process. There are two options here. The first would be to view all word unigram and bigrams strings as coming from the same distribution, ignoring intrinsic differences between the two distributions such as variance. The second option would be to interpolate the two information sources by assigning weights (Lagrange multipliers) to both unigram and bigram contributions, estimating these weights with an estimation procedure. Our hypothesis is that a good balance between these information sources will lead to performance gains.

4.3.3 Classifier setup

After having verified the benefits of the scaled Information Diffusion Kernel (see Section 2.5) in a number of pilot tests, we decided to use this kernel in the experiments reported below. The hyperparameters of the kernel were estimated with the cross-entropy-based beam search algorithm of Section 2.2, using held-out development data. The exact splitting up of the data into training, test and development data is described in Section 4.3.5. By treating the interpolation weights as additional hyperparameters, we estimated these weights jointly with the 'normal' kernel hyperparameters n and t, which seems only natural to do.

Every movie review was indexed for word unigrams and bigrams. The result is a sparse feature vector, consisting of separately normalized frequencies of word unigrams and bigrams. The *hyperkernel* (4.2) was implemented as an extension to the LIBSVM support vector machine toolkit[2].

4.3.4 Related work

Turney [2002] takes an unsupervised approach to polarity classification of doc-

[2]Software available from `http://www.csie.ntu.edu.tw/~cjlin/libsvm`.

uments, by inferring the semantic orientation of documents from the semantic orientation of phrases, which are gathered from the web by neighborhood queries containing positive or negative seed words. Pang et al. [2002], using a predecessor of the movie data of Pang et al. [2002] consisting of 1400 documents, report negative effects on accuracy for polarity classification of documents when using bigrams as the sole features. They note a drop in accuracy of 5.8% for Support Vector Machines applied to the polarity dataset using only bigrams as features. Even the combination of word unigrams and bigrams was outperformed by using unigrams alone. In Pang and Lee [2004], an accuracy of over 87% on the polarity data was achieved by focusing on subjective sentences only. Riloff et al. [2006] find that using both word unigrams and bigrams, in combination with selected complex (syntactic) features taken from an information gain-based subsumption lattice of complex features produces the best results. They achieve 82.7% on the polarity dataset, but they report for 3-fold cross-validation[3] only.

Kennedy and Inkpen [2006] obtain an accuracy of 85.9% on the polarity data (10CV), using Support Vector Machines and valency shifting bigrams, together with word unigrams. Ng et al. [2006] report 90.5% on the polarity data using additional information sources, such as valency information for adjectives, and filter out objective sentences first. To our knowledge, this is the best result reported ever on the polarity data. They report 89.2% using word unigram, bigram and trigram features together, and also notice significant drops in accuracy when using bigrams only. The latter is attributed by the authors to data sparseness: test documents frequently did not contain any of the bigrams in the training documents.

Mullen and Collier [2004] also analyze the movie review data and investigate the combination of various information sources such as semantic orientation and emotive aspects of adjectives, the proximity of emotive phrases to the topic, and lemmatized word unigrams. Using Support Vector Machines, they find that the best results (86% for 10-fold cross-validation for the movie data) are obtained by allocating information sources to separate Support Vector Machines, as opposed to merging them in one feature space and using a single classifier. In their approach, a hyperclassifier combines the model information of the subordinate classifiers, extracting features directly from the lower-level SVM models.

Work on local sentiment prediction, e.g. on the sentence level, is relevant to the work reported in this section when identifying certain combinations of words that are important indicators for the sentiment of a document. Benamara et al. [2007] investigate the use of adverb-adjective combinations for local sentiment analysis. They propose an axiomatic system by which adverbs of degree modify the strength of sentiment of an adjective they combine with. This approach is related to Andreevskaia et al. [2007] and Kennedy and Inkpen [2006], who investigate the use of valency shifting adverbs (like 'not'), that reverse the

[3]See Appendix B.

sentiment of adjectives in their scope. and the work of Wilson et al. [2005]), where polarity expressions are used for computing the polarity of their context. In sum, these lexical approaches, though addressing sentiment on a local level (like sentences), shed light on informative data sources beyond word unigrams for sentiment analysis.

4.3.5 Experiments

We set out to investigate the following questions. First, should we use all word unigrams and bigrams in the training data, or a dedicated selection only? Second, should we interpolate word unigram and bigram information, or just combine these two information sources, pretending they arise from the same distribution? In our experiments, we used the interpolated information diffusion kernel (4.2) and the non-interpolated information diffusion kernel (2.49). For the interpolation experiments, we also investigated the effects of setting one of the weights λ_1 and λ_2 in kernel (4.2) to zero; setting $\lambda_1 = 0$ eliminates the word unigram contribution, and $\lambda_2 = 0$ eliminates the bigram contribution.

4.3.6 Term selection

Following Ng et al. [2006], we ranked all word unigrams and bigrams t in the movie review data based on their weighted log-likelihood ratio (WLLR) scores, for the two classes $c \in \{+1, -1\}$:

$$WLLR(t, c) = P(t \mid c) \ \log \frac{P(t \mid c)}{P(t \mid \neg c)} \tag{4.10}$$

For both classes, we selected the 5,000 highest ranked word unigrams and 5,000 highest ranked word bigrams from the training data as index vocabulary. Some illustrative word unigrams and bigrams from the top 50 WLLR scores are listed in Table 4.2.

4.3.7 Experimental setup

For every experimental condition, we performed 10-fold cross-validation. We permuted the data, and split it into 10 training and test partitions. Every training partition (90% of the data) was subsequently split into a secondary 90% training partition (90%$\times$ 90%=81% of the data), and a 10% development data partition. Finally, every development data partition was subsequently split into a 60% development training partition and a 40% development test partition. The two development data partitionings were used by the hyperparameter estimation algorithm to derive the hyperparameters for the kernel machine, for every separate fold. After the hyperparameter estimation algorithm found these hyperparameters, we trained on the 81% training data partitionings, and tested

Class	unigram
–1	bad, worst, stupid, boring, ridiculous, awful
+1	great, best, well, perfect, wonderful

Class	bigram
–1	this mess, the worst, worst movie a stupid, is terrible, waste of
+1	the best, is excellent, most powerful very effective, a great

Table 4.2: Highly WLLR-ranked word unigrams and bigrams across the two classes.

Interpolation				No interpolation	
	$\lambda_1 \neq 0, \lambda_2 \neq 0$	$\lambda_1 = 0$	$\lambda_2 = 0$		
All terms	**87.3**	79.75	86.2	All terms	80.5
Selected terms	86.4	80.5	84	Selected terms	86.8

Table 4.3: Generalization accuracy (10CV), for the various experimental conditions.

on the corresponding 10% test partitioning. We kept the development data separate from the training data; this makes our results not exactly comparable to 10CV results reported for this data that do not employ development data[4]. Significance of results was measured using a paired t-test.

4.3.8 Results

Table 4.3 lists the results for the various experiments. Results for the interpolated kernel first of all underline the importance of the word unigram contribution. The hyperparameter estimation algorithm predominantly assigned higher weights to the word unigram subkernel across the various folds. The importance of word unigrams becomes clear when manually deactivating the word unigram subkernel, i.e. setting $\lambda_1 = 0$. The drops in accuracy are more dramatic than when eliminating the bigram contribution, sometimes even amounting to over 6%. This is in line with the findings of Pang et al. [2002].

Our results show that the combination of word unigrams and bigrams without performing term selection yields significantly better performance than using only one of these resources, and outperforms term selection followed by either interpolation or no interpolation. The interpolation of all word unigrams and

[4] In fact, comparison of cross-validation results produced by different classifiers on the same dataset should actually be done on exactly the same splits.

word bigrams significantly outperformed the corresponding uninterpolated combination of all terms. When performing term selection, no statistical difference between interpolation or no interpolation could be established; however, these conditions were both outperformed by the interpolation of all terms. We conjecture that the lack of performance gains of interpolation for the selected terms arises from a too aggressive term selection. Clearly, the ad hoc chosen bound of 5,000 highest ranked WLLR terms should be thresholded on development data. The observation that using all terms –including possibly noisy term – benefits the most from interpolation shows that interpolation can stabilize situations of imbalanced information.

4.4 Summary

In this chapter, we proposed a formal method for the combination of heterogeneous sources of information within the multinomial framework, by means of kernel interpolation. We supplied two formal interpretations of kernel interpolation. We subsequently investigated the use of heterogeneous word unigram and bigram information for sentiment classification of movie reviews. We found that a composite kernel consisting of a separate word unigram and bigram kernel, which were interpolated automatically with a cross-entropy-style hyperparameter estimation procedure, produced the best results. These results rank among the best reported using word unigrams and bigrams on the movie review data set of Pang et al. [2002], and are still open to improvement, by incorporating other sources of useful information reported in the literature, such as valency information of adjectives. We conclude that kernel interpolation is a viable technique of practical interest, with a sound formal interpretation.

In Chapter 5, we focus on the problem of language learning with an inherent sequential nature or a very limited amount of information: tasks that do not possess a document structure amenable to frequency-based feature representations.

Chapter 5

Sequential and limited context feature spaces

Up to now we have applied the multinomial framework to document-like objects: full texts (e.g. the polarity classification of entire blog posts and movie reviews) and smaller textual units such as sentences (subjectivity classification) which were expanded into pseudo-documents by using character n-grams. Yet, there are many more classification tasks to be carried out on texts that do not possess this document structure nor are easily reconcilable with character n-gram expansions, specifically linguistic analyses based on small windows of linguistic context. These tasks are essentially ordered. An example is part of speech tagging, where part of speech ambiguities emanating from a lexical lookup step are usually resolved in a limited, left-to-right ordered context[1]. Frequency, the important notion underlying the multinomial framework, is no longer an intuitive notion for these tasks: the frequency of a certain part of speech in the context of an ambiguity to be resolved certainly is interesting information, but we cannot represent each item in the window solely by its relative frequency without losing important information about left-to-right ordering.

In this Chapter, we address the question whether multinomial language learning can indeed be effective here once we switch from counting co-occurrences of features *within a bag* to counting global co-occurrences of features and classes: the estimation of co-occurrence statistics from an entire data set, instead of the strictly local context of a bag of features. In Section 5.1, we outline this idea of *counting outside of the bag*. We present two types of transformations that transform sequential representations into representations suitable to the multinomial classification strategy: first-order and higher-order. The first-order transformation is based on unigram information, whereas the higher-order transformation is based on interpolated unigram and bigram (or higher n-gram) information. We demonstrate that both types of transformation correspond to pullback met-

[1] Part of speech tagging and a number of other ordered tasks are discussed in Daelemans and van den Bosch [2005].

rics of the Fisher information. Experimental results are presented in Section 5.2.

5.1 Co-occurrences of features and classes

In order to apply information diffusion kernels to sequential representations, suitable transformations into a frequency space need to be applied. Co-occurrence of classes and feature values is a logical source of frequency information, expressing characteristics of an entire dataset instead of a small feature space.

5.1.1 First-order models

An apparent candidate L_1-embedding for feature vectors of sequential, multi-class classification problems would be the following transform. Every instance

$$< f_1, \ldots, f_n, c > \tag{5.1}$$

with f_i a feature value, and c a class symbol from the set C of k classes, is replaced by a vector

$$\begin{array}{c} < \frac{|f_1,C_1|}{\sum_{i=1}^{k}\sum_{j=1}^{n}|f_j,C_i|}, \ldots, \frac{|f_1,C_k|}{\sum_{i=1}^{k}\sum_{j=1}^{n}|f_j,C_i|}, \\ \ldots \\ \frac{|f_n,C_1|}{\sum_{i=1}^{k}\sum_{i=1}^{n}|f_j,C_i|}, \ldots, \frac{|f_n,C_k|}{\sum_{i=1}^{k}\sum_{i=1}^{n}|f_j,C_i|}, c > \end{array} \tag{5.2}$$

This effectively is an L_1-embedding of joint observations of feature values and classes. We shall refer to this type of representation as *co-occurrence transform*. Notice that the transform in (5.2) ignores positional information of features. For this reason, we refer to it as a *position-unaware transform*. A *position-aware* version would be

$$\begin{array}{c} < \frac{|f_1[1],C_1|}{\sum_{i=1}^{k}\sum_{j=1}^{n}|f_j[j],C_i|}, \ldots, \frac{|f_1[1],C_k|}{\sum_{i=1}^{k}\sum_{j=1}^{n}|f_j[j],C_i|}, \\ \ldots \\ \frac{|f_n[n],C_1|}{\sum_{i=1}^{k}\sum_{i=1}^{n}|f_j[j],C_i|}, \ldots, \frac{|f_n[n],C_k|}{\sum_{i=1}^{k}\sum_{i=1}^{n}|f_j[j],C_i|}, c > \end{array} \tag{5.3}$$

where $f_i[j]$ means *feature value* f_i *at position* j. An example will make the difference clear.

Ratnaparkhi [1998b] presents a task known as PP-attachment, consisting of the attachment of a prepositional phrase as either a noun or a verb modifier, on the basis of a four-cell window containing a verb, a noun, a preposition, and a noun. An example from this task is

- `cast, brothers, as, brothers, V`

This example states that the prepositional phrase *as brothers* is a modifier of the verb *cast* rather than the noun *brothers*. The capitalized 'V' is a class symbol; the second class symbol for this binary classification task is 'N'. The position-unaware transform (5.2) would transform this instance to

$$
\begin{array}{l}
< \dfrac{|cast,N|}{\begin{array}{l}|\ cast,N\ |\ +\\ |\ cast,V\ |\ +\\ |\ brothers,N\ |\ +\\ |\ brothers,V\ |\ +\\ |\ as,N\ |\ +\\ |\ as,V\ |\ +\\ |\ brothers,N\ |\ +\\ |\ brothers,V\ |\end{array}}, \dfrac{|cast,V|}{\begin{array}{l}|\ cast,N\ |\ +\\ |\ cast,V\ |\ +\\ |\ brothers,N\ |\ +\\ |\ brothers,V\ |\ +\\ |\ as,N\ |\ +\\ |\ as,V\ |\ +\\ |\ brothers,N\ |\ +\\ |\ brothers,V\ |\end{array}}, \dfrac{|brothers,N|}{\begin{array}{l}|\ cast,N\ |\ +\\ |\ cast,V\ |\ +\\ |\ brothers,N\ |\ +\\ |\ brothers,V\ |\ +\\ |\ as,N\ |\ +\\ |\ as,V\ |\ +\\ |\ brothers,N\ |\ +\\ |\ brothers,V\ |\end{array}}, \dfrac{|brothers,V|}{\begin{array}{l}|\ cast,N\ |\ +\\ |\ cast,V\ |\ +\\ |\ brothers,N\ |\ +\\ |\ brothers,V\ |\ +\\ |\ as,N\ |\ +\\ |\ as,V\ |\ +\\ |\ brothers,N\ |\ +\\ |\ brothers,V\ |\end{array}},\\
\dfrac{|as,N|}{\begin{array}{l}|\ cast,N\ |\ +\\ |\ cast,V\ |\ +\\ |\ brothers,N\ |\ +\\ |\ brothers,V\ |\ +\\ |\ as,N\ |\ +\\ |\ as,V\ |\ +\\ |\ brothers,N\ |\ +\\ |\ brothers,V\ |\end{array}}, \dfrac{|as,V|}{\begin{array}{l}|\ cast,N\ |\ +\\ |\ cast,V\ |\ +\\ |\ brothers,N\ |\ +\\ |\ brothers,V\ |\ +\\ |\ as,N\ |\ +\\ |\ as,V\ |\ +\\ |\ brothers,N\ |\ +\\ |\ brothers,V\ |\end{array}}, \dfrac{|brothers,N|}{\begin{array}{l}|\ cast,N\ |\ +\\ |\ cast,V\ |\ +\\ |\ brothers,N\ |\ +\\ |\ brothers,V\ |\ +\\ |\ as,N\ |\ +\\ |\ as,V\ |\ +\\ |\ brothers,N\ |\ +\\ |\ brothers,V\ |\end{array}}, \dfrac{|brothers,V|}{\begin{array}{l}|\ cast,N\ |\ +\\ |\ cast,V\ |\ +\\ |\ brothers,N\ |\ +\\ |\ brothers,V\ |\ +\\ |\ as,N\ |\ +\\ |\ as,V\ |\ +\\ |\ brothers,N\ |\ +\\ |\ brothers,V\ |\end{array}}\\
,V >
\end{array}
\tag{5.4}
$$

The position-aware transform would produce a similar result, with one difference: all counts would now be based on feature-class co-occurrences taking positional information into account:

$$
\begin{array}{l}
< \dfrac{|cast[1],N|}{\begin{array}{l}|\ cast[1],N\ |\ +\\ |\ cast[1],V\ |\ +\\ |\ brothers[2],N\ |\ +\\ |\ brothers[2],V\ |\ +\\ |\ as[3],N\ |\ +\\ |\ as[3],V\ |\ +\\ |\ brothers[4],N\ |\ +\\ |\ brothers[4],V\ |\end{array}}, \dfrac{|cast[1],V|}{\begin{array}{l}|\ cast[1],N\ |\ +\\ |\ cast[1],V\ |\ +\\ |\ brothers[2],N\ |\ +\\ |\ brothers[2],V\ |\ +\\ |\ as[3],N\ |\ +\\ |\ as[3],V\ |\ +\\ |\ brothers[4],N\ |\ +\\ |\ brothers[4],V\ |\end{array}}, \dfrac{|brothers[2],N|}{\begin{array}{l}|\ cast[1],N\ |\ +\\ |\ cast[1],V\ |\ +\\ |\ brothers[2],N\ |\ +\\ |\ brothers[2],V\ |\ +\\ |\ as[3],N\ |\ +\\ |\ as[3],V\ |\ +\\ |\ brothers[4],N\ |\ +\\ |\ brothers[4],V\ |\end{array}}, \dfrac{|brothers[2],V|}{\begin{array}{l}|\ cast[1],N\ |\ +\\ |\ cast[1],V\ |\ +\\ |\ brothers[2],N\ |\ +\\ |\ brothers[2],V\ |\ +\\ |\ as[3],N\ |\ +\\ |\ as[3],V\ |\ +\\ |\ brothers[4],N\ |\ +\\ |\ brothers[4],V\ |\end{array}},\\
\dfrac{|as[3],N|}{\begin{array}{l}|\ cast[1],N\ |\ +\\ |\ cast[1],V\ |\ +\\ |\ brothers[2],N\ |\ +\\ |\ brothers[2],V\ |\ +\\ |\ as[3],N\ |\ +\\ |\ as[3],V\ |\ +\\ |\ brothers[4],N\ |\ +\\ |\ brothers[4],V\ |\end{array}}, \dfrac{|as[3],V|}{\begin{array}{l}|\ cast[1],N\ |\ +\\ |\ cast[1],V\ |\ +\\ |\ brothers[2],N\ |\ +\\ |\ brothers[2],V\ |\ +\\ |\ as[3],N\ |\ +\\ |\ as[3],V\ |\ +\\ |\ brothers[4],N\ |\ +\\ |\ brothers[4],V\ |\end{array}}, \dfrac{|brothers[4],N|}{\begin{array}{l}|\ cast[1],N\ |\ +\\ |\ cast[1],V\ |\ +\\ |\ brothers[2],N\ |\ +\\ |\ brothers[2],V\ |\ +\\ |\ as[3],N\ |\ +\\ |\ as[3],V\ |\ +\\ |\ brothers[4],N\ |\ +\\ |\ brothers[4],V\ |\end{array}}, \dfrac{|brothers[4],V|}{\begin{array}{l}|\ cast[1],N\ |\ +\\ |\ cast[1],V\ |\ +\\ |\ brothers[2],N\ |\ +\\ |\ brothers[2],V\ |\ +\\ |\ as[3],N\ |\ +\\ |\ as[3],V\ |\ +\\ |\ brothers[4],N\ |\ +\\ |\ brothers[4],V\ |\end{array}},\\
,V >
\end{array}
\tag{5.5}
$$

For the position-unaware transform, the counts $|\ brothers, N\ |$ and $|\ brothers, V\ |$ are not based on the actual position of 'brothers' in the feature vectors in the training data, as opposed to the position-aware transform.

For binary tasks like this, the feature-class co-occurrences are complementary, e.g.

$$|\ cast, V\ | = 1 - |\ cast, N\ | \tag{5.6}$$

so the output of the transform can be simplified by leaving out one of the probabilities.

In practice, many events ($|\ f_i, c\ |$ or $P(c\ |\ f_i)$) in the test data will be unobserved in the training data, but simple smoothing methods can be applied to solve this problem, such as Good-Turing and Laplacian add-one methods (Gale and Sampson [1995]).

5.1.1. PROPOSITION. *The co-occurrence transforms are pullback metrics of the Fisher information.*

Proof Let θ_i denote the L1-normalized frequency of the i-th feature, and c a class; f_i is the value of the i-th feature, and C_j is the j-th class. Starting with the original sequence of L1-normalized feature frequencies $\theta_1, \ldots, \theta_n$ (which in the sequential case are of the form $\frac{1}{n}$), we see that the embedding obtained by the co-occurrence transforms is a pullback metric of the Fisher information:

$$G_\lambda(\theta) = \left(\frac{\theta_i \lambda_i^1}{\sum_{j=1}^{k} \sum_{i=1}^{n} \theta_i \lambda_i^j}, \ldots, \frac{\theta_i \lambda_i^k}{\sum_{j=1}^{k} \sum_{i=1}^{n} \theta_i \lambda_i^j}, \ldots, \frac{\theta_n \lambda_n^1}{\sum_{j=1}^{k} \sum_{i=1}^{n} \theta_i \lambda_i^j}, \ldots, \frac{\theta_n \lambda_n^k}{\sum_{j=1}^{k} \sum_{i=1}^{n} \theta_i \lambda_i^j} \right) \tag{5.7}$$

where $\lambda_i^j = \ n \cdot \mid f_i, C_j \mid$, or $\lambda_i^j = \ n \cdot P(C_j \mid f_i)$. ■

Adding an extra subscript generalizes this result to the position-aware transform.

5.1.2 Higher-order models

First-order models are based on collocations of feature values (eventually on a certain position in a feature vector) and classes. Higher-order models would take into account contextual information, like collocations of sequences of features and classes. A combination of these two approaches is possible, and common practice in the language modeling field. It is possible to construct a back-off model using Expectation-Maximization (Dempster et al. [1977]) that estimates two or more models differing in complexity, and a set of interpolation parameters estimated from held-out development data. The interpolation parameters determine the relative weight of the different sources of information. Estimating an interpolation weight for a separate information source is in fact related to hyperkernel interpolation, as it also consists of uniform weighting of an entire feature (information) space.

In particular, it is possible to back off from a joint frequency model that addresses co-occurrences of longer sequences of features and classes, to a model of co-occurrences of shorter sequences of features and classes. Given a sequence of n features $f_1, \ldots, f_n$, let f_i^k be the subsequence of features $f_i \ldots f_{i+k}$. If $k > n$, then f_i^k is the empty sequence ϵ, and for all classes $c \in C$, $P(c \mid \epsilon) = 0$.

Algorithm C.4 in Appendix C is an instantiation of the EM-algorithm that can be used to smooth L1-normalized conditional probabilities, by interpolating a-th order probabilities (based on joint occurrences of a features and classes) with b-th order probabilities ($a > b$). Every λ_k is the value of the interpolation

parameter λ at time k, which is used to interpolate the two models M_a and M_b. These λ's can be computed from the training data.

Log-likelihood of the data given the estimated parameters at time k is defined as

$$L(f_1^n; \lambda_k) = \sum_i \sum_{c \in C} \log(\lambda_k P_a(c \mid f_i^a) + (1 - \lambda_k) P_b(c \mid f_i^b)) \tag{5.8}$$

A suitable stopping criterion for Algorithm C.4 in Appendix C can consist of a test on change of log-likelihood, based on a threshold θ:

$$L(f_1^n; \lambda_k) - L(f_1^n; \lambda_{k-1}) < \theta \tag{5.9}$$

or, similarly, a test on change of the λ parameter:

$$\lambda_k - \lambda_{k-1} < \theta \tag{5.10}$$

For a k-class problem and two models M_a and M_b, the original feature vectors $< f_1, \ldots, f_n, c >$ are replaced by

$$< \frac{\sum_{c \in C} \lambda P_a(c|f_1^a) + (1-\lambda) P_b(c|f_1^b)}{\sum_{c \in C} \sum_i^n \lambda P_a(c|f_i^a) + (1-\lambda) P_b(c|f_i^b)}, \ldots, \frac{\sum_{c \in C} \lambda P_a(c|f_n^a) + (1-\lambda) P_b(c|f_n^b)}{\sum_{c \in C} \sum_i^n \lambda P_a(c|f_i^a) + (1-\lambda) P_b(c|f_i^b)} > \tag{5.11}$$

5.1.2. Proposition. *The higher-order co-occurrence transform can be reformulated as a pullback metric of the Fisher information.*

Proof The proof of the proposition is direct, by the following:

$$G_\lambda(\theta; a, b) = \left(\frac{\theta_i \lambda_i^{a,b}}{\sum_{i=1}^n \theta_i \lambda_i^{a,b}}, \ldots, \frac{\theta_n \lambda_n^{a,b}}{\sum_{i=1}^n \theta_i \lambda_i^{a,b}} \right) \tag{5.12}$$

where $\lambda_i^{a,b} = n \cdot \left(\sum_{c \in C} \lambda P_a(c \mid f_i^a) + (1 - \lambda) P_b(c \mid f_i^b) \right)$. ■

5.2 Experiments

In order to verify the usefulness of co-occurrence-based pullback metrics for sequential tasks, we experimented with the following sequential tasks:

- English prepositional phrase attachment (PP; Ratnaparkhi [1998b], and briefly introduced in Section 5.1.1).
- German plural noun formation (GPLURAL; Daelemans and van den Bosch [2005]), the classification of a noun to one out of eight plural conjugation classes, on the basis of a 7-feature window. The 7 features describe the contents of the two final syllables of the noun and its grammatical gender.

- Dutch diminutive formation (DIMIN; Daelemans and van den Bosch [2005]), the prediction of a diminutive suffix (5 classes) for Dutch nouns on the basis of its last three syllables, which are described by 4 features each: the presence or absence of stress of the syllable, its onset (start), nucleus (middle part) and coda (ending).

- English shallow parsing (CHUNK; Daelemans and van den Bosch [2005]), the shallow parsing of sentences (WSJ Penn Treebank) into major constituents (22), such as NP, VP, PP, on the basis of a 7-cell window containing words and parts of speech. The tagging is based on the `IOB` schema (Ramshaw and Marcus [1995]), where `I-XP` indicates the presence of a word inside a phrase `XP`, `O-XP` indicates being outside a phrase `XP`, and, `B-XP` indicates the start of a phrase `XP` preceded by another `XP` of the same category.

We used only a part of CHUNK: the first 10,000 first items in the training partition, and the first 4,000 items in the test partition for testing. These datasets have fixed training/test partitions. Statistics are listed in Table 5.1, as well as the results obtained with memory-based learners for these datasets, as reported by Daelemans and van den Bosch [2005][2]. We report their highest scores and indicate the type of measure. Scores not reported are indicated with a hyphen.

In our experiments, we varied the use of pullback metric: position-aware ('pos') and position-unaware ('no-pos'). We evaluated a maximum entropy representation ('maxent') consisting of L1-normalized 'count once' frequencies (see Section 3.1.4), which can be seen as a baseline representation or this type of learning tasks, where feature repetition (and hence the occurrence of frequencies greater than 1) is minimal. This implies that for this representation, the use of positional information is not relevant: counting takes place *inside the bag* and every feature-position combination receives a count of 1, normalized to $\frac{1}{n}$ for n features. We further evaluated the interpolation between unigram features ($P(c \mid f_i)$) and bigram features ($P(c \mid f_i f_{i+1})$), using the position-aware pullback transform.

5.3 Results

Our experiments demonstrate that co-occurrence transforms are effective for all tasks as compared with the maximum entropy representations. Although for PP only improvement in F_{micro} is observed, the average of F_{micro} and F_{macro} for the position-aware pullback is 81.9, vs. 81.6 for the maximum entropy representation. For PP and DIMIN, the pullback transforms obtain higher results than the results reported by Daelemans and van den Bosch [2005]. For

[2]These authors reported sometimes accuracy, sometimes F-scores (and not both).

Task	# Train	# Test	# Classes	Accuracy	F_1
PP	20,801	3,097	2	80.7	–
GPLURAL	12,584	12,584	6	94.3	–
DIMIN	2,999	950	5	97.6	–
CHUNK	211,727	47,377	22	–	91.9

Table 5.1: Statistics of the sequential tasks. The accuracy and F-scores, obtained with memory-based learners, are reported by Daelemans and van den Bosch [2005].

GPLURAL, the interpolated classifier improves on the result reported by these authors. For CHUNK, our results cannot be easily compared, but the micro-averaged F1-score obtained by the position-aware pullback suggests comparable performance.

The position-aware pullback metric uniformly outperforms the position-unaware pullback, as is to be expected for sequential tasks. As can be seen, interpolation is not effective for PP, CHUNK and DIMIN, but it is effective for GPLURAL. For this dataset, the computed interpolation parameter was $\lambda = .47$. For the other datasets this parameter was much higher, showing there is virtually no information to be gained from the bigrams, except for CHUNK, where the situation is opposite: the low value of $\lambda = .1$ indicates that the unigrams are virtually uninformative. For GPLURAL, unigrams and bigrams happen to be roughly equally informative. Since their contribution apparently is complementary, the interpolation strategy is able to improve here on the simple pullback transform.

5.4 Related work

We are not aware of work in the multinomial language learning tradition that addresses sequential and limited context tasks. Interpolation of several sources of information for sequential tasks has a longstanding tradition, e.g. Brants [2000] (part-of-speech tagging) and Black et al. [1993], Collins [2003] (parsing).

Hall and Hofmann [2000] investigate the use of multinomial submanifolds for the purpose of dimension reduction for n-gram language models. This work, predating geodesic kernel methods on the multinomial simplex, does not address quasi-document structures. Extensions of multinomial classification methods that model sequential information (like topic segmentation) instead of static word frequency-based information have been presented by Lebanon [2006b]. These methods operate on document level, by fitting a kernel to model changes in word frequencies under a bag-of-words approach (the *lowbow* approach).

Numeric transformations from discrete data based on conditional probabili-

Task	Condition	Accuracy	F_{micro}	F_{macro}
No interpolation				
PP	maxent	**82.9**	80.6	**82.6**
	L1,no-pos	80.6	80.6	80.3
	L1,pos	82.1	**82.1**	81.7
GPLURAL	maxent	88	88	79.6
	L1,no-pos	90.5	90.5	82.1
	L1,pos	**91.9**	**91.9**	**85.3**
DIMIN	maxent	79.2	79.2	57.7
	L1,no-pos	96.8	96.8	89.2
	L1,pos	**97.8**	**97.8**	**92.4**
CHUNK	maxent	38.3	38.3	10.2
	L1,no-pos	69.1	69.1	22.3
	L1,pos	**91.5**	**91.5**	**39.1**
Interpolation				
PP	$\lambda = .89$	79	80.5	80.5
GPLURAL	$\lambda = .47$	**94.6**	**94.6**	**91**
DIMIN	$\lambda = .73$	95	95	80.3
CHUNK	$\lambda = .1$	80	80	26.2

Table 5.2: Experimental results for the sequential tasks. Best results are indicated with boldface.

ties were proposed by Kasif et al. [1998], who devised a numeric transform based on the MVDM, or Modified Value Difference Metric proposed by Stanfill and Waltz [1987]:

$$< f_1, \ldots, f_n, c > \mapsto < P(+1 \mid f_1), \ldots, P(+1 \mid f_n) > \text{ for } c \in \{-1, +1\} \quad (5.13)$$

in the context of k-nearest neighbor classification. Their distance metric essentially being Euclidean, this L_1-embedding of feature spaces does not benefit from, nor pay attention to the intrinsic geometry of the resulting multinomial data.

5.5 Summary

In this chapter, we have proposed two types of transformations that allow for the application of multinomial classifiers to sequential and limited context tasks.

We presented first-order and higher-order transformations that respectively address unigram and higher n-gram information. By computing feature-class co-occurrence statistics, we solve the problem of having insufficient information for applying multinomial classifiers to limited context, sequential learning tasks. We demonstrated that the proposed transforms formally correspond to pullback metrics of the Fisher information, thus providing the approach with a sound formal basis. Our experimental results show that the 'position-aware' pullback metric –taking into account the position a feature value occurs on, when computing feature-class co-occurrence statistics– is the most effective for these tasks. An interpolation strategy providing a back-off option to a unigram model was found to be particularly effective for one task, where the interpolation parameter computed showed roughly equal informativity of unigram and bigram features.

This brings us to the end of Part I of this thesis, where we have both verified the performance of standard multinomial classifiers (Section 3) and proposed a number of extensions in order to deal with heterogeneous information in a principled manner (Section 4) and sequential, limited context tasks (Section 5). In Part II, we will assess exactly under which conditions multinomial classifiers with geodesic distance measures perform well, and under which conditions Euclidean distance would be a better option.

Part II

A Back-Off Model for Document Geometry

Chapter 6

Isometry, entropy and distance

In this chapter we first prove in Section 6.1 analytically that for maximum entropy data, the Euclidean and negative geodesic distance measures become isometric. Then, we expose a particular weakness of geodesic distance measures based on the inverse cosine: certain regions in the domain of the arccos function produce either rounding errors or non-linear behavior. We identify the conditions under which these phenomena take place, which have important ramifications for the applicability of geodesic kernels. Subsequently, Section 6.3 proposes a least squares based method for inspecting the geometry of data.

6.1 An algebraic perspective on isometry

In this section we prove the existence of a mapping from K_{NGD} to K_{EUCLID} and vice versa for maximum entropy data. This shows that for this type of data, K_{EUCLID} and K_{NGD} display *metric isometry* (Lee, 1997, p.112), by the existence of a homeomorphism between them. Two metrics $D1$ and $D2$ on a manifold M are topologically equivalent or isometric if the identity mapping

$$I : (M, D1) \mapsto (M, D2) \tag{6.1}$$

is bijective, continuous, and has a continuous inverse. Let (M, K) be a manifold endowed with a metric K, and let M_{L1}^{ME} be a maximum entropy L1-normalized vector space. Then we need to show that

$$I : (M_{L1}^{ME}, K_{EUCLID}) \mapsto (M_{L1}^{ME}, K_{NGD}) \tag{6.2}$$

and, vice versa,

$$I : (M_{L1}^{ME}, K_{NGD}) \mapsto (M_{L1}^{ME}, K_{EUCLID}) \tag{6.3}$$

6.1.1. PROPOSITION. *L1-normalized maximum entropy data endowed with K_{NGD} is isometric to Euclidean space.*

Proof Given two vectors $\mathbf{x}$ and $\mathbf{y}$, let $\mathbf{x_1}$ ($\mathbf{y_1}$) be the number of dimensions in which $\mathbf{x}$ ($\mathbf{y}$) has a non-zero value:

$$\mathbf{x_1} =\mid x_i \neq 0 \mid \tag{6.4}$$

Similarly, let $\mathbf{x_0}$ ($\mathbf{y_0}$) be the number of dimensions in which $\mathbf{x}$ ($\mathbf{y}$) has a zero value:

$$\mathbf{x_0} =\mid x_i = 0 \mid \tag{6.5}$$

Let a (*agreements*) be

$$a =\mid x_i = y_i \mid \tag{6.6}$$

and let d (*differences*) be a quantity composed of

$$\begin{array}{l} d_1 =\mid x_i \neq y_i, x_i \neq 0, y_i \neq 0 \mid \\ d_2 =\mid x_i \neq y_i, x_i = 0 \mid \\ d_3 =\mid x_i \neq y_i, y_i = 0 \mid \end{array} \tag{6.7}$$

Finally, let c (*collisions*) be the number of components in which both $\mathbf{x}$ and $\mathbf{y}$ simultaneously have a non-zero value:

$$\begin{array}{ll} c = & \mid x_i = y_i \mid + \mid x_i \neq y_i \mid -\mathbf{x_0} - \mathbf{y_0} = \\ & a + d - \mathbf{x_0} - \mathbf{y_0} = \\ & a + d_1 + d_2 + d_3 - \mathbf{x_0} - \mathbf{y_0} \end{array} \tag{6.8}$$

The Euclidean distance metric (Section 2.3) then computes the following quantity:

$$\begin{array}{ll} K_{EUCLID}(\mathbf{x}, \mathbf{y}) = & \sqrt{d_1 \left(\frac{1}{\mathbf{x_1}} - \frac{1}{\mathbf{y_1}}\right)^2 + d_2 \left(\frac{1}{\mathbf{y_1}}\right)^2 + d_3 \left(\frac{1}{\mathbf{x_1}}\right)^2} = \\ & \sqrt{\frac{d_1}{\mathbf{x_1}^2} - \frac{2d_1}{\mathbf{x_1 y_1}} + \frac{d_1}{\mathbf{y_1}^2} + \frac{d_2}{\mathbf{y_1}^2} + \frac{d_3}{\mathbf{x_1}^2}} = \\ & \sqrt{\frac{d_1+d_3}{\mathbf{x_1}^2} - \frac{2d_1}{\mathbf{x_1 y_1}} - \frac{d_1+d_2}{\mathbf{y_1}^2}} \end{array} \tag{6.9}$$

For K_{NGD}, we have

$$\begin{array}{ll} K_{NGD}(\mathbf{x}, \mathbf{y}) = & -2 \arccos\left(c\sqrt{\left(\frac{1}{\mathbf{x_1}} \cdot \frac{1}{\mathbf{y_1}}\right)}\right) = \\ & -2 \arccos\left(\frac{a+d_1+d_2+d_3-\mathbf{x_0}-\mathbf{y_0}}{\sqrt{\mathbf{x_1} \cdot \mathbf{y_1}}}\right) \end{array} \tag{6.10}$$

We can now re-express K_{NGD} as a function of K_{EUCLID}:

$$K_{NGD'}(\mathbf{x},\mathbf{y}) =$$

$$-2\arccos\left(\left(\left(K_{EUCLID}(\mathbf{x},\mathbf{y})^2 + \frac{d_1+d_2}{\mathbf{y_1}^2} - \frac{d_1+d_3}{\mathbf{x_1}^2}\right) \times -\frac{\frac{a}{d_1}+1+\frac{d_2}{d_1}+\frac{d_3}{d_1}-\frac{\mathbf{x_0}}{d_1}-\frac{\mathbf{y_0}}{d_1}}{2}\right) \times \frac{1}{\sqrt{\mathbf{x_1 y_1}}}\right)$$

$$= -2\arccos\left(\left(\left(K_{EUCLID}(\mathbf{x},\mathbf{y})^2 + \frac{d_1+d_2}{\mathbf{y_1}^2} - \frac{d_1+d_3}{\mathbf{x_1}^2}\right) \times -\frac{a+d_1+d_2+d_3-\mathbf{x_0}-\mathbf{y_0}}{2d_1}\right) \times \frac{1}{\sqrt{\mathbf{x_1 y_1}}}\right) \tag{6.11}$$

Likewise

$$K_{EUCLID'}(\mathbf{x},\mathbf{y}) =$$

$$\sqrt{(\left(\left(\cos\frac{K_{NGD}(\mathbf{x},\mathbf{y})}{-2}\right) \times \frac{\sqrt{\mathbf{x_1 y_1}}}{1}) \times -\frac{2}{\frac{a}{d_1}+1+\frac{d_2}{d_1}+\frac{d_3}{d_1}-\frac{\mathbf{x_0}}{d_1}-\frac{\mathbf{y_0}}{d_1}}\right) + \frac{d_1+d_3}{\mathbf{x_1}^2} - \frac{d_1+d_2}{\mathbf{y_1}^2}}$$

$$= \sqrt{\left(\left(\cos\frac{K_{NGD}(\mathbf{x},\mathbf{y})}{-2}\right) \times \sqrt{\mathbf{x_1 y_1}} \times -\frac{2d_1}{a+d_1+d_2+d_3-\mathbf{x_0}-\mathbf{y_0}}\right) + \frac{d_1+d_3}{\mathbf{x_1}^2} - \frac{d_1+d_2}{\mathbf{y_1}^2}} \tag{6.12}$$

■

The mapping $K_{NGD'}$ maps K_{EUCLID} to K_{NGD}, and the inverse mapping $K_{EUCLID'}$ maps K_{NGD} to K_{EUCLID}: the operations involved are exactly paired, and each operation in $K_{EUCLID'}$ is the inverse of an operation in $K_{NGD'}$. Note that we would never be able to relate K_{EUCLID} and K_{NGD} for non-maximum entropy data, as all probabilities 'vanish' in the aggregate product $\sum_i \sqrt{x_i y_i}$ that is part of K_{NGD}, and we can't eliminate the square root operation. This is easiest to show if we take the linear dot product as a Euclidean distance measure. Suppose we want to map the NGD kernel to this product. As a first step,

$$\cos\left(\frac{K_{NGD(\mathbf{x},\mathbf{y})}}{-2}\right) = \sum_{i=1}^{n} \sqrt{\mathbf{x_i y_i}} \tag{6.13}$$

But now there is no possibility of eliminating the various squared products by means of exponentiation, as these are all combined in one product.

The operations in (6.12) are bijective, continuous mappings. From the existence of these mappings it becomes clear that the two distance metrics coalesce (only) in the case of maximum entropy L1-normalized data. Yet there is a difference between these two metrics under maximum entropy, as we will point out in the next subsection.

6.2 The relation between entropy and distance

In Section 6.3 we investigate empirically under which conditions this approximation becomes particularly close. Prior to that, we notice the following. Let D be a set of K documents $d_1 \ldots d_n$, $n \geq 1$. Any document $d \in D$ is represented as a bag (a set with repetition) of words, which together make up the vocabulary of d, V_d. A *frequency variant* d' of d would be any element of the set $\{d' \in V_d^* \mid d' \supseteq d\}$. For instance, the document $a\ a\ b\ b\ c\ c$ is a frequency variant of the document $a\ b\ c$. Let $\mathcal{L}$ be the total set of L1-normalized possible frequency variants of D. The *Gram matrix* $\mathcal{G}_{l,D,M}$ is the matrix of pairwise distances between all elements of D, under the representation l and the geodesic distance measure $K_{NGD} \subseteq D \times D \mapsto \mathbb{R}$ The average Gram distance $\overline{\mathcal{G}}_{l,D,M}$ is defined as $\frac{1}{K^2} \sum_{i=1}^{n} \sum_{j=1}^{n} \mathcal{G}_{l,D,M}(i,j)$. Then the following holds:

6.2.1. Proposition. $\forall l' \in \mathcal{L} :\ if\ H(l') \leq H(\arg\max_{l \in \mathcal{L}} H(l))\ then\ \overline{\mathcal{G}}_{l',D,M} \geq \overline{\mathcal{G}}_{l,D,M}$

Proof The proposition follows from the observation that when $x \approx 1$, $\arccos(x) \approx 0$, and $-2\arccos(x) \approx 0$, and when $x \approx 0$, $\arccos(x) \approx \frac{1}{2}\pi$, and $-2\arccos(x) \approx -\pi$. Under a maximum entropy situation, for two n-dimensional vectors x and y, probabilities x_i, y_i converge to $\frac{1}{n}$. The sum of squared probability products then approaches

$$\sum_{i=1}^{n} \sqrt{x_i y_i} \approx \sum_{i=1}^{n} \sqrt{\frac{1}{n^2}} \approx \sum_{i=1}^{n} \frac{1}{n} \approx 1$$

■

for quite similar vectors; any feature that is defined in vector θ but not in vector θ' (or vice versa) will produce a zero contribution to the total sum of probabilities.

This proposition states that, under a maximum entropy representation, objects become maximally close to each other in the representing manifold: the maximum entropy representation yields the smallest possible average distance between objects, since values close to 1 of the dot product that is part of the NGD kernel lead to distances around zero.

6.2.2. Remark. It is well known that the arccos function, part of the NGD kernel, becomes inaccurate for small values due to large rounding errors (Sinnott [1984]). Figure 6.1 displays the rounding error for $\mid x - \cos(\arccos(x)) \mid$, for $x \in [0, 1]$. While the function composition $\cos \circ \arccos$ should in theory produce the identity function, in practice, on current hardware, application of arccos introduces a rounding error. For small values of x, this rounding error is relatively high, and becomes zero for values of x above 0.7. These small

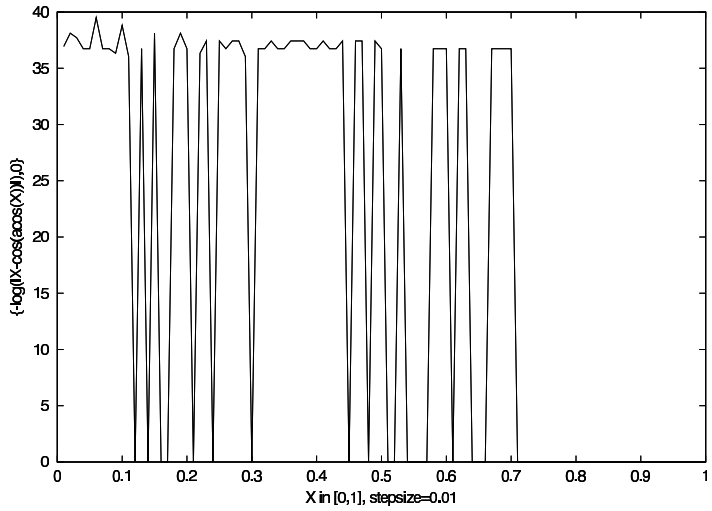


Figure 6.1: Rounding error for $\cos(\arccos(x))$ with $x \in [0, 1]$. The y-axis plots $\mid x - \cos(\arccos(x)) \mid$ on a log scale. Differences between x and $\cos(\arccos(x))$ equal to zero receive $y = 0$.

values occur easily for maximum entropy data when overlap between vectors is minimal. Notice further that $\arccos(0.9) = 0.45$, and that $\arccos(0.99) = 0.14$, where $\arccos(1) = 0$, which shows a non-linear steep descent behavior of arccos for values close to 1. See Figure 6.2, which illustrates this trend for arguments above 0.7, by showing a fragment of the poor result of a linear least squares fit, with large inaccuracy especially for the value range $[0.7, 1]$. This non-linearity will lead to unjustifiably large differences between vectors that differ only in a few features.

We surmise that this causes distance measures based on arccos to work suboptimally for low similarity values, as well as for values close to 1. We summarize this for clarity.

- High entropy data increases the risk for dot products with small values for highly dissimilar vectors; for these small values the NGD kernel produces distances close to $-\pi$, with large rounding errors.
- High entropy data increases the risk for dot products close to 1 for very similar vectors; for these large values the NGD kernel produces distances close to zero, with non-linear behavior.

Returning to Proposition 6.1.1, for maximum entropy cases, the NGD kernel still is based on the inverse cosine (Eqn. 6.11), whereas the Euclidean kernel

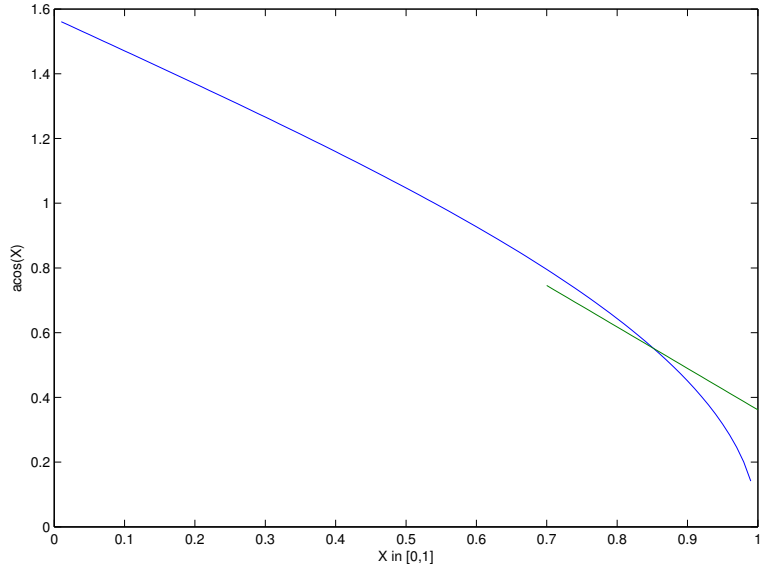


Figure 6.2: The function $\arccos(x)$ for $x \in [0, 1]$, and a least squares fit, shown here only for the range $[0.7, 1]$.

applies cos to the NGD kernel (Eqn. 6.12). This reformulation reduces the NGD kernel to a rounding error-free dot product for values of the dot product approaching 1, where $\cos(\arccos(x)) \approx x$:

$$\begin{gathered} \cos(\tfrac{K_{NGD(\mathbf{x},\mathbf{y})}}{-2}) = \\ \cos(\arccos \textstyle\sum_i \sqrt{x_i y_i}) = \\ \textstyle\sum_i \sqrt{x_i y_i} \end{gathered} \tag{6.14}$$

So, for the boundary case of maximum entropy, the reformulated Euclidean kernel effectively *compiles away* the arccos function, which favors this kernel over the reformulated NGD kernel by eliminating arccos-based errors. Therefore, under high to maximum entropy conditions, it would make sense to use Euclidean kernels that omit the arccos function. In Chapter 8 we will re-address this issue and put forward a procedure that thresholds the arccos-based error component of the NGD kernel. Also, we will propose a novel kernel based on the so-called haversine distance (Sinnott [1984]), that is particularly accurate on small distances.

6.3 Least squares estimation of isometry

In the previous section we showed an algebraic isometry between K_{NGD} and K_{EUCLID} for maximum entropy L1-normalized data. In this section, we develop a pragmatic method that, for a given L1-normalized dataset, will determine if there indeed is isometry between K_{EUCLID} and K_{NGD}.

6.3.1. PROPOSITION. *A Riemannian manifold M_1 is isometric to a Euclidean manifold M_2 if there exists a constant c such that the Gram matrix for M_1 is equal to c times the Gram matrix for M_2.*

Proof Let D_{M_1}, D_{M_2} be the associated distance metrics with M_1 and M_2. By definition, if there exists a homeomorphism h mapping D_{M_1} to D_{M_2}, then M_1 and M_2 are isometric (Lee [1997]). Clearly, $h(D_{M_1}) = c \cdot D_{M_2}$ is bijective, continuous, and has a continuous inverse. ■

Since multiplication with a constant is a linear operation, we can use linear methods such as Least Squares Estimation (LSE) to determine the existence of such a constant. The least squares method minimizes the sum of squared residuals, i.e.

$$E = \sum_{i=1}^{n} \left[y_i - f(x_i, \beta) \right]^2 \tag{6.15}$$

where β is a set of parameters, y_i the dependent or observed variable, x_i a predicted value, and $f(x, \beta)$ a model function relating x_i to y_i through the parameters $\beta_1, \ldots, \beta_m$:

$$f(x, \beta) = \sum_{i=1}^{m} f'_j(x)\beta_j \tag{6.16}$$

Let r_i denote the *residual* $[y_i - f(x_i, \beta)]$. Minimizing E corresponds to setting its gradient to zero:

$$\frac{\partial E}{\beta_j} = 2 \sum_{i}^{n} r_i \frac{\partial r_i}{\partial \beta_j} = 0 \tag{6.17}$$

which is equivalent to

$$-2 \sum_{i}^{n} r_i \frac{\partial f(x_i, \beta)}{\partial \beta_j} r_i = 0 \tag{6.18}$$

For the linear case this can be expressed as

$$-2 \sum_{i}^{n} f'_j(x_i) \left(y_i - \sum_{k}^{m} f'_j(x_i)\beta_k \right) = 0 \tag{6.19}$$

Given two distance metrics $D1$ and $D2$ on a certain manifold M, we can compute the Gram matrices $G1$ and $G2$. The Gram matrices contain exactly

n^2 distances for n datapoints. If the two metric spaces $(M, D1)$ and $(M, D2)$ are isometric, then the LSE problem $G1 \cdot z = G2$ should have a solution, and therefore we can carry out the computations specified in Algorithm C.5.

Typically, $E_G$in Algorithm C.5 will be around zero in the case of full isometry. In practice, due to rounding errors and numerical instability, the error will be over zero. Normalization can be applied to two respective Gram errors $E1, E2$ to compute the *relative Gram error*

$$G_R(E_1 \mid E_2) = \frac{E_1}{E_1 + E_2} \tag{6.20}$$

If the least squares method outlined above finds an isomorphism (a Gram error ≈ 0), then the K_{NGD} kernel is no more effective than the K_{EUCLID} kernel.

Experiment

In order to assess the usefulness of this procedure, we performed the following experiment. We used the AMI sentiment dataset (Carletta et al. [2005]), a dataset consisting of 13 speech recognized scenario-based meeting recordings, of approximately 30 minutes each. Every meeting had four participants. This dataset was annotated by Wilson [2008] for both sentiment subjectivity (the presence or absence of an opinion in an utterance) and polarity (the classification of an opiniated utterance as positive, negative or neutral). In this data, utterances consist of dialogue act segments (see Raaijmakers et al. [2008] and Raaijmakers and Wilson [2008]). A subjective utterance is a sequence of words where a *private state* is being expressed. A private state (Quirk et al. [1985]) is an an opinion, belief, sentiment, emotion, etc. The subjectivity problem is a two-class problem. On average, every meeting contains about 1150 manually transcribed speech utterances, with roughly 107,723 (65.5%) objective and 56,628 (34.5%) subjective utterances. This is a difficult task that is best modeled with features that capture sequential information (e.g. Raaijmakers and Kraaij [2008]).

For every meeting of the AMI sentiment dataset, we took 10 random samples of 500 datapoints (40% on average) from the training file of that particular fold, for both the normal L1-representation and the maximum entropy L1-representation. For each type of representation, the same random sample was used. Subsequently, we computed the Euclidean and NGD Gram matrices for each of these 130 500 points samples, after which the Gram error was measured. The average Gram error of the maximum entropy L1-data was .008% of the average Gram error obtained with the normal L1-data (equivalently, the L1-based Gram error was 124 times higher than the maximum entropy-based Gram error). So, the relative Gram error of the maximum entropy data compared to the original L1 data was effectively around zero, and the relative Gram error for the normal L1 data was close to 1. The maximum entropy L1-normalized

data therefore was much more similar to Euclidean data than the normal L1 data: the differences between the two errors were found to be highly significant by a Wilcoxon ranked sum ($p < .002$).

6.4 Summary

In this chapter we have both analytically and empirically verified that the entropy of L1-normalized data has immediate consequences for the application of geodesic kernels. We demonstrated analytically that for maximum entropy L1-normalized data, the NGD and Euclidean distance metrics are isometric. As a corollary, we argued that high entropy data is represented by regions on a manifold with small distances between points. In fact, maximum entropy data produces the smallest distances possible between data points. For high entropy data with a substantial degree of overlap between vectors, dot products between vectors approach 1, which leads to non-linear behavior of the arccos function part of the NGD kernel. Small differences lead to significant perturbations of the outcome of arccos. For vectors with a limited amount of overlap, dot products produce small values for which arccos produces large rounding errors. We conclude that the NGD kernel does not yield optimal performance on these boundary cases. The Euclidean kernel, while isometric to the NGD kernel for maximum entropy data, compiles away the arccos function for maximum values of dot products, omitting the non-linear behavior of arccos in the high end of its domain. Finally, we proposed a diagnostic method that determines isometry of the NGD and Euclidean kernels through least squares computations on the respective Gram matrices. We showed that maximum entropy data is much better represented with a Euclidean Gram matrix than with a geodesic Gram matrix.

In Chapter 7, we will provide additional evidence challenging the point of view that L1-normalized data should be uniformly analyzed with geodesic distance measures, from a *performance* point of view: we will carry out a number of experiments that assess the performance of geodesic and Euclidean distance measures when confronted with high entropy data.

Chapter 7

Hybrid geometry

In Chapter 6, we have established analytically a relationship between distance and entropy. In this chapter, we provide additional empirical evidence for the hypothesis that, under certain circumstances, L1-normalized document space possesses hybrid Euclidean-geodesic geometry. Section 7.1 argues that dimensionality reduction (or feature selection), when based on hybrid geometrical principles, is able to outperform single-geometry methods on certain data, while this situation is reversed on other data. Likewise, Section 7.2 presents a manifold denoising strategy based on combined Euclidean and geodesic geometry that uniformly outperforms single-geometry denoising. These observations show that L1-normalization does not automatically entail the unconditional use of geodesic distance measures.

7.1 Dimensionality reduction

In this section, we investigate the use of geometric feature selection approaches to document dimensionality reduction. If L1-normalized documents indeed are naturally embedded in Riemannian manifolds, then it makes sense to look for dimensionality reduction techniques that operate exactly on these manifolds. On the other hand, if documents have both geodesic and Euclidean properties, then a mixed approach might make sense. In this section we will outline and evaluate exactly such an approach.

The contributions of this section are the following. First, we propose a feature weighting algorithm that, using a geometric method, performs a blockwise analysis of input data, where feature weights are computed for every data block. These feature weights are averaged after completion of the algorithm, and are used to select features. We compare standard Euclidean Laplacian Eigenmaps with a proposed variant: geodesic Laplacian Eigenmaps. The latter uses the distance function inherent to Riemannian manifolds. In addition, we evaluate a combination of these two, where the weights computed by the Euclidean Eigen-

map and the geodesic Eigenmap are averaged. This variant combines information from two representational spaces: the Euclidean space and the Riemannian manifold. Evaluation of the weighting algorithms is performed by deleting iteratively low-weight features. We demonstrate for two text classification tasks that the geodesic variant offers the best performance, and, for experiment 2, that the combined approach significantly outperforms the other approaches, providing evidence for the hypothesis that hybrid geometrical representations of documents may be beneficiary.

7.1.1 The curse of dimensionality

As most text classification problems suffer from the *curse of dimensionality* – a large, sparse feature space, and an abundance of irrelevant or noisy features – feature selection is an active area of research in the text classification community. A feature selection algorithm attempts to identify valuable features and eliminates features whose presence is detrimental for the performance of a classifier on a certain dataset. A great variety of feature selection techniques have been proposed throughout the years – including analytical approaches (e.g. information gain-based filtering (e.g. Yang and Pedersen [1997], Forman [2003]), heuristic methods (e.g. wrapper-based methods (Kohavi and John [1997]; Raaijmakers [1999])) and search-based methods (e.g. genetic algorithms (the early work of Siedlecki and Sklansky [1989], and, more recently, Daelemans et al. [2003]) and ant colony search algorithms (e.g. Aghdam et al. [2009]). A particular class of feature selection algorithms consists of methods that take into account the underlying *geometry* of the data. Modeling the data as an information space (a topological space, or manifold) with a certain dimensionality (the number of features), these methods aim at reducing high-dimensional information spaces to lower dimensional subspaces with minimal loss of information. Well-known examples are PCA, LLE and Isomap (for an overview see der Maaten [2007] and der Maaten [2009]). So-called *Laplacian Eigenmaps* have been proposed by Belkin and Niyogi [2002] and Gerber et al. [2007] as efficient methods for dimensionality reduction of manifold structures. A Laplacian Eigenmap reconstructs a neighborhood graph approximating the geodesic distances in the original manifold, and projects this neighborhood to a lower-dimensional manifold that faithfully represents the original higher-dimensional data and preserves local neighborhoods. The neighborhood reconstruction is based on Euclidean distance; the neighborhood graph consists of weighted edges connecting nearest neighbors. The computation of the low-dimensional manifold Y from an original manifold X involves minimizing a cost function C:

$$C(Y) = \sum_{ij} (y_i - y_j)^2 W_{ij} \tag{7.1}$$

with W_{ij} the weight of the edge connecting points i and j. Given a suitable choice of weights, the cost function can be made to produce a penalty when the original points x_i, x_j are mapped too far apart in the reconstruction Y. From the weight matrix W, the diagonal matrix weight M (the row sums of $W, M_{ij} = \sum_j W_{ij}$) can be computed. The graph Laplacian is $L = M - W$. It encodes a graph as a matrix by listing the difference between the degree matrix (the number of incoming nodes, for any node) and the adjacency matrix (the number of adjacent nodes, for any node)[1]:

$$L_{ij} = \begin{cases} deg(vertex_i) \; if \; i \neq j \\ -1 \; if \; i \neq j \; and \; vertex_i \; is \; a \; neighbor \; of \; vertex_j \\ 0 \; else \end{cases}$$

The cost function can be re-expressed as

$$C(Y) = 2Y^T LY \tag{7.2}$$

Minimizing C becomes equivalent to solving

$$L\mathbf{v} = \lambda M\mathbf{v} \tag{7.3}$$

where the d smallest nonzero eigenvalues constitute the low-dimensional embedding Y:

$$\begin{aligned} L\mathbf{v}_0 &= \lambda_0 M\mathbf{v}_0 \\ &\vdots \\ L\mathbf{v}_{d+1} &= \lambda_{d+1} M\mathbf{v}_{d+1} \end{aligned} \tag{7.4}$$

The low-dimensional embedding Y thus consists of the mapping

$$Y : x_i \mapsto (\mathbf{v}(i)_1, \ldots, \mathbf{v}(i)_m) \tag{7.5}$$

The value of m can be found automatically as the *intrinsic dimensionality* of the data (der Maaten [2007]). Algorithm C.6 in Appendix C summarizes these steps.

The generic Laplacian Eigenmap algorithm solves the problem of dimensionality reduction for general manifolds. Euclidean distance is used as a generic distance measure for the construction of the neighborhood graph (Belkin and Niyogi [2002]):

$$|| \, x_i - x_j \, ||^2 \tag{7.6}$$

This approach thus essentially makes a similar assumption about the geometry of the input data space: this space is assumed to be Euclidean, and distances

[1] Notice that the graph Laplacian is neutral with respect to eventual negative distances in the neighborhood graphs arising from a negatively shifted kernel such as K_{NGD}: as long as the neighborhood relations are preserved (by measuring absolute differences between data points), the actual value of the distances is of no importance to the computation of the graph Laplacian.

are best measured along straight lines connecting datapoints. In the present case, as we focus on L1-normalized textual data, we propose a geodesic variant of Laplacian Eigenmaps that uses the negative geodesic distance (K_{NGD}) for the computation of nearest neighbors.

7.1.2 Feature weighting and selection with Eigenmaps

The products of the eigenvectors and the eigenvalues produced by the Laplacian Eigenmaps can be used as feature weights (der Maaten [2007]). They describe the importance of the weight of the original dimensions for the reconstruction process: the higher the weight, the more important the original dimension (or: the feature). Due to the large dimensionality of feature selection problems, we propose a blockwise approach to finding these weights. Given a data matrix D with r rows and c columns, we perform feature weighting on all submatrices $d = D(r_i : r_j, c_k : c_l)$ We call $r_j - r_i$ the row step size, and $c_l - c_k$ the column step size. Operating row-first, the algorithm averages the weights for every pass through all row blocks. Effectively, it constructs Laplacian Eigenmaps for subspaces, and combines the weight information that arises from this process. The final weight vector is a matrix with one row and c columns.

An example will make this clear. Given a 10x10 matrix D, first construct a Laplacian Eigenmap for the submatrix $d_1(1:5, 1:5)$, then for $d_2(5:10, 1:5)$, then average the weight matrices for d_1 and d_2, $W1$ and $W2$:

$$W_1 \dots W_5 = \frac{1}{2} \left(W1_1 \dots W_5 + W2_5 \dots W2_{10} \right) \tag{7.7}$$

Then proceed with the submatrices $d_3(1:5, 5:10)$ and $d_4(5:10, 5:10)$, producing $W3$ and $W4$. The final weight vector W then becomes

$$[\frac{1}{2} \left(W1_1 \dots W1_5 + W2_5 \dots W_{10} \right), \frac{1}{2} \left(W3_1 \dots W3_5 \right) + W4_5 \dots W4_{10})] \tag{7.8}$$

This weight averaging procedure is outlined in Algorithm C.7. Algorithm C.9 computes a one-row n-column feature weight vector, by traversing blockwise through a data matrix with n columns. It uses the COMPUTE-EIGENVALUES algorithm (Algorithm C.8) that returns the entire matrix of eigenvalues[2].

The matrix is split up into submatrices, each of which is subject to dimension reduction. The weights emerging from this process are averaged over all submatrices.

The parameters that have to be estimated are the row and column step size, and the number of k for the neighborhood computations by the Eigenmaps.

[2]See Appendix C for these algorithms.

	Task A	Task B
Number of labeled emails	4,000	100
Number of emails within one evaluation/tuning inbox	2,500	400
Number of inboxes for evaluation	3	15
Number of inboxes for tuning	1	2

Table 7.1: The ECML 2006 Discovery Challenge data.

Hybrid weights

In order to verify the existence of hybrid geometry in documents, we propose to combine the weights from a Euclidean and geodesic Laplacian Eigenmap, and compare the effects to either Euclidean or geodesic Eigenmaps. Probably the simplest combination strategy consists of a simple weight averaging scheme. For a data set (matrix) M,

$$W = \frac{1}{2}[\lambda_1 W_{NGD} + \lambda_2 W_{EUCLID}] \tag{7.9}$$

where W_{NGD} and W_{EUCLID} are found by BLOCKWISE-FEATURE-WEIGHTING using respectively δ_{NGD} and δ_{EUCLID} as distance measures. We report results for uniform choices of $\lambda_1, \lambda_2 = 1$. Further, as described below, we will evaluate the feature weighting process by iteratively deleting low-weight features, thus performing feature selection based on feature weights.

7.1.3 Data

Our first dataset consists of the 13 sentiment-annotated meetings from the AMI Meeting Corpus (Section 6.3). The second dataset we use is a spam detection dataset, available from the ECML 2006 Discovery Challenge[3]. The challenge addresses the separation of spam from non-spam e-mail messages, and consists of two separate tasks: a task (A) with user-specific training data, addressing user-specificity, and a task (B) with a limited amount of data per user, addressing generalization over users. All data sets consist of word/frequency pairs, which were L1-normalized. The average entropy for this data is around 6 bits. The breakdown in terms of training and test data is listed in Table 7.1. We used task A, where for three users (mailboxes) there are 4000 labeled training email messages and 2500 test cases. Like experiment 1, this experiment is a two-class problem (spam vs. non-spam).

All data was L1-normalized and no feature construction nor data cleanup was performed; using all words the data thus consists of L1-normalized unigram

[3] http://www.ecmlpkdd2006.org/challenge.html

frequencies. We have made no effort to obtain absolute high scores for these tasks, as we are interested in relative performance only.

7.1.4 Experimental setup

Our classifier uses the negative geodesic kernel K_{NGD} discussed in Section 2.5.2. which we implemented in LIBSVM. The Eigenmap feature selection algorithms were implemented as modifications of and extensions to the Dimensionality Reduction Toolkit of der Maaten [2007][4].

For experiment 1, we set apart two meetings (two folds) as development data for parameter estimation. On this data, we estimated the row and column stepsize and the k value for the Eigenmap neighborhood function. Observing no major differences between choices for these values, we settled on stepsizes of 10% and 1% for the rows and columns, and $k = 10$. The other 11 folds were used for 11-fold cross-validation: in every round of the testing procedure, one meeting was used for testing, and the other ten meetings were used for training the classifier and for constructing the feature map. The data has a dimensionality of 3,784 features.

For experiment 2, we ran into a practical limitation of Matlab® 7: the number of variables (dependent on the number of features) became too big to handle. The training data for this task has a dimensionality of 206,908, which we limited to 10,000 (the dimensionality of the data of experiment 1) by eliminating every feature with an index (based on occurrence in the data) above 10,000. The fact that we are interested mainly in relative performances legitimates this approach[5]. We used the same k-value, and row/column stepsize as for experiment 1.

For both experiments, we ran the standard Euclidean Eigenmap (M_{EUCLID}) and the geodesic Eigenmap (M_{NGD}). The weights produced by these algorithms were ordered, and starting with all features, portions of 10% were eliminated (the lightest features were eliminated first). Subsequently, the two feature maps produced were averaged, and a similar feature elimination was carried out based on the mixed Euclidean/geodesic weights (M_{COMBI}).

7.1.5 Evaluation

We evaluate the experimental outcomes using subjective, macro-averaged F_1 and AUC (area under curve) scores, for all folds and partitionings of the datasets. Specifically, we averaged the F1 and AUC scores for all folds per feature selection percentage. On these averaged scores, we computed significance with a non-parametric Wilcoxon signed rank test.

[4] Software available from `http://ticc.uvt.nl/~lvdrmaaten`.

[5] In the latest Matlab® release, limitations with respect to the number of variables are lifted.

From a usability perspective, an optimal feature weighting and selection algorithm would produce a performance curve where the maximum performance is reached with a minimum of features: it achieves maximal compression with minimal loss of performance. We define the following simple error measure for a performance curve $S =< s_1, \ldots, s_n >$, where every s_i is the score of a certain performance function for a measurement at time tick i, $max(S)$ the maximum value in S, and σ_n the standard deviation among the top most n values in S.

$$E(S,n) = \min(\begin{array}{l} [\arg\min_{s_i} s_i \leq max(S) \& \\ max(S) - s_i \leq \sigma_n], \\ \arg\min_{s_j} s_j \geq \max(S) \\ ) \end{array} \tag{7.10}$$

This error function finds the first good operating point in a sequence of ordered measurements that either is above the maximum value in S, or at most one standard deviation less than this maximum, where the standard deviation is measured among the top n of the best scores in S. The compression rate of a feature selection algorithm A producing a performance curve S can then be defined as

$$CR_A(S,n) = 1.0 - E(S,n) \tag{7.11}$$

7.1.6 Results and discussion

For experiment 1 results show that the geodesic feature selector M_{NGD} consistently reaches its first good operating point before the Euclidean and mixture selectors do: its compression rate is 10% higher than for the Euclidean feature selector EUCLID. Figure 7.1 and Figure 7.2 display the average area under curve (AUC) and F1 scores for the 11 folds.

At around 60% of the features, M_{NGD} reaches its optimum. This means that 40% of the features were eliminated without loss in performance. For M_{EUCLID}, this operating point was only reached at 70%: only 30% could de eliminated without loss in performance. Table 7.2 lists the averaged AUC and F1 results. While the M_{NGD} feature selector outperformed or equalled M_{EUCLID} and M_{COMBI} in terms of error scores, significance of these results could not be established at $p < .05$. In Table 7.3, the error scores for the three feature selectors are listed for the 4 performance curves. With the exception of AUC, M_{NGD} outperforms the other two selectors, reaching its first good operating point around 60% of all features.

For experiment 2 however, we see that M_{NGD} consistently and significantly outperforms M_{EUCLID}. The averaged combination of M_{NGD} and M_{EUCLID} even outperforms significantly both M_{NGD} (F1) and M_{EUCLID} (AUC and F1) (see Table 7.4). Also, it can be observed that feature selection is quite advantageous for this task. The combined system obtains a maximum AUC score

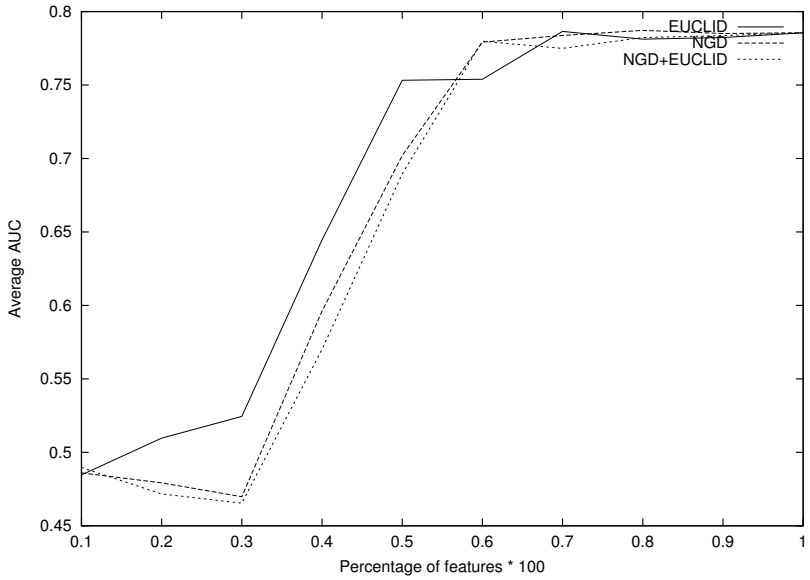


Figure 7.1: AUC curve for experiment 1.

at 30% of the features that is 10% higher than when using all features. Likewise, a peak score for F1 is obtained at 60% of the features that is 11% higher than when using all features. The fact that averaging the weights from the Euclidean and NGD-based Eigenmaps produces better performance indicates that this data is best modeled partially by Euclidean space and partially by a Riemannian manifold.

7.1.7 Related work

Lebanon [2003] is concerned with finding low-dimensional Riemannian manifolds or textual data. In his work, the problem is to learn an optimal metric on Riemannian manifolds. Metric candidates are pullback metrics of the Fisher information that maximize the inverse volume of the training data. In doing so, they de-emphasize common words and emphasize more distinctive terms, just like $tf.idf$. This process can, like $tf.idf$, be interpreted as a feature weighting method.

Zhao et al. [2008] presents a feature scoring algorithm based on Laplacian Eigenmaps. Their intuition is that important features preserve the local geometry of the original data in the reconstructed submanifold. Their Laplacian Score is based on a weighted quotient of the Laplacian weight matrix and its diagonal. They report high compression rates and effectiveness on the Iris dataset and

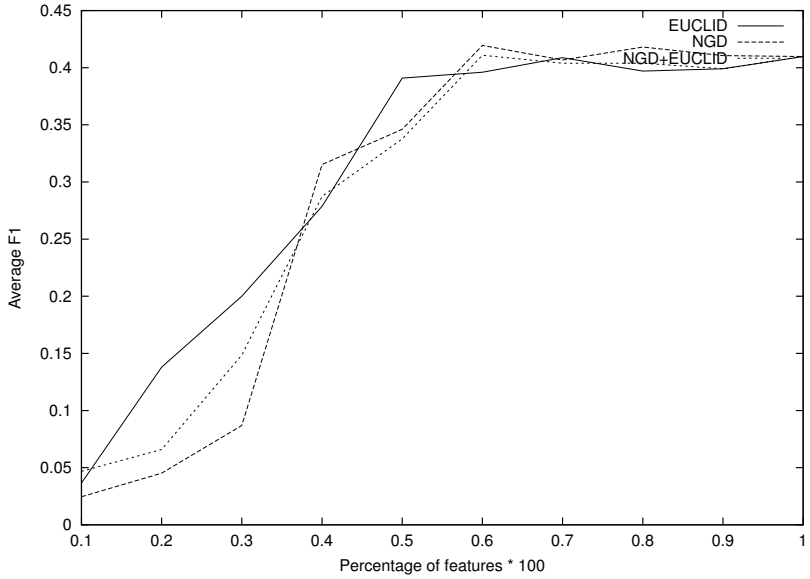


Figure 7.2: F1 curve for experiment 1.

PIE face data set.

In Lu et al. [2007] a variant of PCA called Principle Feature Analysis (PFA) is proposed. Feature vectors are decomposed into vectors (one for every feature) that represent the weight of every feature on each axis of a subspace. These vectors can be clustered, and the mean element of each cluster becomes selected as a feature. Their approach, applied to image data, uses a similar blockwise subspace decomposition as our approach.

Guérif and Bennani [2007], drawing inspiration from weighted k-means clustering, use Self-Organizing Maps (SOM; Kohonen [2001]) that deploy a weighted distance function which replaces the standard unweighted Euclidean distance function used for computing neighborhoods. This weighted distance function can be optimized by the standard SOM algorithm.

Gerber et al. [2007] identify a fundamental problem of non-linear dimensionality reduction of noisy data, the *repeated eigendirections problem*. This problem arises when eigenvectors computed from the graph Laplacian are strongly correlated. This hinders the discovery of important dimensions of the low-dimensional manifold. Their algorithm, Successive 1-Dimensional Laplacian Eigenmaps, reconstructs iteratively one eigenvector at a time (the one with the smallest non-zero eigenvalue), and the dimension found by this eigenvector is eliminated from subsequent computations. The approach is effective, yet computationally demanding.

	Average	M_{NGD}	M_{EUCLID}	M_{COMBI}
AUC				
M_{NGD}	66.5	X	=	+
M_{EUCLID}	68.1	=	X	=
M_{COMBI}	65.9	–	=	X
F1				
M_{NGD}	28.8	X	=	=
M_{EUCLID}	30.6	=	X	=
M_{COMBI}	29.1	=	=	X

Table 7.2: Average AUC and F1 outcomes of experiment 1, and significance results ($p < 0.05$).

$E(S, n = 3)$	M_{NGD}	M_{EUCLID}	M_{COMBI}
F1	0.6	0.7	0.6
AUC	0.8	0.7	1
Recall	0.6	0.7	1
Precision	0.6	0.7	0.6

Table 7.3: Error scores for experiment 1

The work by Underhill et al. [2007] is closely related to our work, in that it investigates a number of dimensionality reduction techniques such as PCA and Isomap applied to *tf.idf* encoded documents. The methods they use do not deploy geodesic distance measures. They find that using quite simple methods (i.c. Multi-Dimensional Scaling (MDS) and Isomap) perform the best in yielding significant compression ratios and accuracy gains during classification.

We are, to the best of our knowledge, not aware of geometrical feature selection approaches using neighborhood graphs that operate on multinomial representations for documents.

7.2 Manifold denoising

In this section, we attempt to gather additional evidence for the possibility of hybrid geometry of L1-normalized documents from the perspective of manifold denoising, a technique aiming at eliminating noisy data from a manifold. Manifold denoising techniques rely on distance measures. We develop a geodesic distance-based denoising algorithm, and compare it to a standard Euclidean version and a hybrid Euclidean-geodesic version. Our hypothesis is that a care-

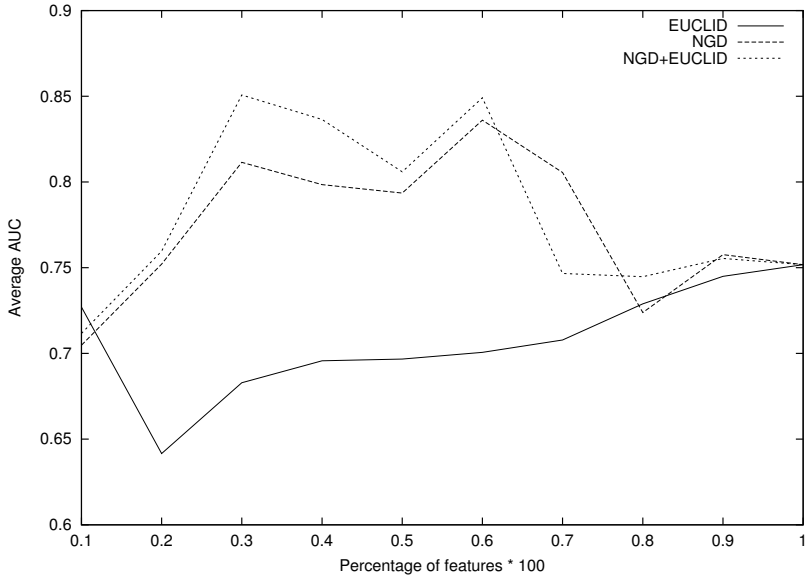


Figure 7.3: AUC curve for experiment 2.

ful comparison of these three denoising algorithms sheds light on the geometry of the noise in a manifold, and, hence, on the geometry of the manifold itself.

7.2.1 Diffusion-based manifold denoising

Manifold denoising attempts to project noisy data in a manifold back onto a low-dimensional, noise-free submanifold that intrinsically represents the data. The data, as represented by data points in a high dimensional feature space, is obscured by noise that hides the inherent manifold structure representing the data. This noise could originate from artifacts in the input data, such as features with low predictive quality, from annotation errors, or from accidentally skewed data that misrepresents the actual problem. The challenge is to effectively eliminate this noise and project the data onto the hidden submanifold. This presupposes a noise model, and a means of restructuring the manifold in such a way that noise interacting with the structure of the manifold is eliminated as much as possible. Hein and Maier [2007] propose a method that assumes the noise is isotropic or Gaussian, and put forward an algorithm that iteratively removes this noise. An arbitrary manifold M of dimension m is assumed to be mappable via a smooth function i into its actual, 'true' manifold of dimension d ($d \leq m$):

$$i : M \mapsto \mathbb{R}^d \tag{7.12}$$

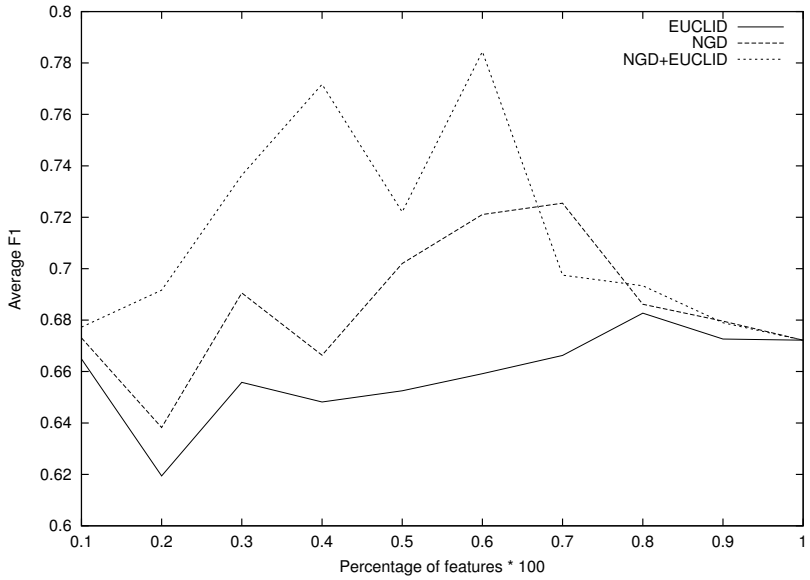


Figure 7.4: F1 curve for experiment 2.

A dataset X can then be described as 'true' data (generated by a probability measure P_M) obscured by noise ϵ:

$$X = i(P_M) + \epsilon \tag{7.13}$$

The probability measure P_M is the true model of the data in M. It can be used to estimate the density of the noisy data X (Hein and Maier [2007]):

$$P_X(x) = (2\pi\sigma^2)^{\frac{d}{2}} \int_M e^{-\frac{\|x-i(\theta)\|^2}{2\sigma^2}} p(\theta) dV(\theta) \tag{7.14}$$

where dV, the natural volume element, is an estimate of the volume of a manifold on the basis of the metric associated with it.

The well-known *heat kernel* (Lafferty and Lebanon [2005]) is equivalent to this formulation under the condition $\sigma^2 = 2t$. It plays an important role in the modeling of information (or temperature) diffusion processes in networks. Viewing noise generation as a diffusion process, Hein and Maier propose to model noise elimination as a *backward* diffusion process. If we knew P_X and P_M, noise elimination would be trivial. In practice, we only have available a partial characterization of P_X: a sample $\{X_{i=1}^n\}$. The crucial insight, coming from Hein et al. [2005], is that the data sample $\{X_i\}$ is sufficient, as the true generator of the diffusion process, a graph Laplacian (Section 7.1.1), can even be approximated by a graph Laplacian of a random neighborhood graph.

	Average	M_{NGD}	M_{EUCLID}	M_{COMBI}
AUC				
M_{NGD}	77.4	X	+	=
M_{EUCLID}	70.8	–	X	–
M_{COMBI}	**78.1**	=	+	X
F1				
M_{NGD}	68.6	X	+	–
M_{EUCLID}	65.9	–	X	–
M_{COMBI}	**71.3**	+	+	X

Table 7.4: Average AUC and F1 outcomes of experiment 2, and significance results ($p < 0.05$).

$E(S, n = 3)$	M_{NGD}	M_{EUCLID}	M_{COMBI}
F1	0.1	0.8	0.1
AUC	0.2	0.9	0.2
Recall	0.9	0.6	1
Precision	0.1	0.1	0.1

Table 7.5: Error scores for experiment 2

Algorithm C.10 describes the original Hein and Maier algorithm for manifold denoising. The crucial ingredient is the adaptation of all data vectors at time step $t + 1$ as a function of the graph Laplacian at time t and the parameter δt. The algorithm gradually moves noisy datapoints towards high-density areas. The data changes during this process; as such, the approach is related to the competitive learning strategy found in self-organizing maps (Kohonen [2001]). Notice that the Hein and Maier algorithm is based on Euclidean distance measurements in the neighborhood graph for the data. A geodesic variant is presented in Algorithm C.11. It differs from the original algorithm by replacing the Euclidean distance function h with the geodesic distance function δ_{GD}, defined as

$$\delta_{GD}(\mathbf{x}) =\mid K_{NGD}(\mathbf{x}, \mathbf{y}) \mid= K_{GD}(\mathbf{x}, \mathbf{y}) = 2 \arccos\left(\sum_{i=1}^{n} \sqrt{x_i y_i}\right) \quad (7.15)$$

($\mathbf{y}$ *the nearest neighbor of* $\mathbf{x}$)

We investigate whether this geodesic variant is better suited for eliminating noise than the Euclidean version, for L1-normalized data. In order to verify this, we need an objective function that measures (quantifies) the effect of noise

elimination on data. As the result of manifold denoising with the Hein and Maier algorithm is a nearest neighbor graph, we implement an objective function as a transductive nearest neighbor classifier derived from the output graph, and evaluate its performance on test data. We compare the original version of the Hein and Maier algorithm with a special geodesic version, as well as a hybrid version.

For voting purposes, the geodesic distances, taking their values in the interval $[\pi, 0]$ (with π for the most dissimilar cases) were normalized to the interval $[0, 1]$ with the following formula

$$\delta \mapsto \mid \frac{\delta}{\pi} - 1 \mid \tag{7.16}$$

This leads to a weight of zero for the dissimilar cases, and a weight of 1 for the most similar cases. Following Hein and Maier, we subsequently treat these weights exponentially using $exp(-\gamma\delta)$, where we uniformly used $\gamma = 1$, producing $exp(-\delta)$.

The combined data (unlabeled training and test data) is used to construct the nearest neighbor graph that encodes the denoised manifold. Subsequently, a distance-weighted nearest neighbor classifier is extracted from the nearest neighbor graph, that uses the class information of the nearest neighbors that correspond to training data for voting over the class of each test point. This effectively amounts to a form of *transductive learning*, where feature information of the test data (with class labels kept hidden) is used during learning. While not pretending to produce an optimal classifier, this strategy does allow us to evaluate the effects of the two types of noise elimination with an objective function. Optimal parameters for the denoising algorithms were estimated by grid search on heldout development data. This gives rise to Algorithms C.12 (the evaluation of single-geometry manifold denoising with a transductive classifier) and Algorithm C.13 (the evaluation of hybrid-geometry manifold denoising with a transductive classifier).

In order to evaluate these alternatives, we carried out a number of *Monte Carlo tests*. Our data consisted of the datasets PP, GPLURAL and DIMIN, introduced in Section 5.2. In addition, we used the NER dataset from the CoNLL-2003 shared task (Tjong Kim Sang and Meulder [2003]), a 4-class named entity recognition task (persons, organizations, locations, miscellaneous) using windows of 7 words and their parts of speech, annotated in the IOB schema (Section 5.2).

From each of the datasets PP, GPLURAL, DIMIN and NER, we took a random portion of 6,000 items, which we permuted 5 times, and from which, for every permutation, we split off 5,000 items for training and development, and 1,000 for testing. As to our data representation, we used the class-feature co-occurrence pullback of Chapter 5. We subsequently applied the dual method transductive algorithm. The algorithm was optimized on the development data,

using a grid search across its parameter space. Keeping record of the intermediate single-method results allowed us to check the effect of merging Euclidean and geodesic nearest neighbors sets on classifier performance. Notice that we are not pursuing optimal performance compared to a baseline. We are merely gathering evidence for the hypothesis that, for certain data, hybrid geometry is a more natural representation, as opposed to uniform Euclidean or geodesic geometry. Therefore, we will only report relative performances.

7.2.2 Results

In Table 7.6, the macro-averaged F-scores for the four tasks are listed. Table 7.7 lists the results of a paired t-test to the obtained results. For PP, the combined classifier geo+euclid significantly outperforms the single-source geodesic classifier, but not the single source Euclidean classifier. Apparently, the combination benefits most from the Euclidean neighbors. For GPLURAL, this situation is reversed: the combined classifier significantly outperforms the Euclidean single source classifier, but not the geodesic classifier. The combined classifier attains the absolute highest F-score, and the overlap between nearest neighbor sets is minimal compared to the other tasks. For NER, the Euclidean single source classifier supersedes the geodesic classifier, and for DIMIN, the geodesic single source classifier outperforms the Euclidean classifier. For these last two tasks, the combined classifier is not able to improve upon the single source classifiers.

Task	Classifier	macro-F	% NN Overlap
PP	geo	85.1	58
	euclid	85.5	
	geo+euclid	**85.7**	
GPLURAL	geo	67.8	6
	euclid	65.9	
	geo+euclid	**69.7**	
NER	geo	48.3	29
	euclid	**52**	
	geo+euclid	50.2	
DIMIN	geo	**81.2**	25
	euclid	77	
	geo+euclid	**81**	

Table 7.6: Macro-averaged F-score, for four tasks, obtained with transductive classifiers.

Spearman's ranked correlation test reveals a strong negative correlation ($\rho = -.8$) between the amount of overlap of the nearest neighbor sets and

the difference between the F-score produced by the combined nearest neighbors and the averaged single-source F-scores. The higher the overlap between the Euclidean and geodesic neighbor sets, the lower the performance gain of the combined classifiers. Notice that this effect is not attributable to voting over a larger space of nearest neighbors: the optimal value for k, the size of the nearest neighborhoods, was already determined during the classifier optimization step. A strong geodesic classifier will boost a weak Euclidean classifier, and vice versa, but not if their nearest neighbors overlap considerably. This leads us to Observation 7.2.1:

7.2.1. Observation. *The extent to which a dataset has mixed Euclidean and geodesic geometry determines the efficacy of single-source nearest neighbor classifiers.*

PP			
	geo	euclid	geo+euclid
geo	X	=	- ($p < .03$)
euclid	=	X	=
geo+euclid	+ ($p < .03$)	=	X
GPLURAL			
	geo	euclid	geo+euclid
geo	X	=	=
euclid	=	X	- ($p < .6$)
geo+euclid	=	+ ($p < .6$)	X
NER			
	geo	euclid	geo+euclid
geo	X	- ($p < .01$)	- ($p < .03$)
euclid	+ ($p < .01$)	X	+ ($p < .01$)
geo+euclid	+ ($p < .03$)	- ($p < .01$)	X
DIMIN			
	geo	euclid	geo+euclid
geo	X	+ ($p < .04$)	=
euclid	- ($p < .04$)	X	- ($p < .05$)
geo+euclid	=	+ ($p < .05$)	X

Table 7.7: Significance results of the F-scores according to a paired t-test, for four transductive classifiers.

7.3 Summary

In this chapter, we have motivated along two empirical lines the possibility of hybrid geometry in L1-normalized document space. In Section 7.1, we presented an effective and low-complexity feature selection algorithm based on Laplacian Eigenmaps. Focusing on L1-normalized textual data lying on a Riemannian manifold, we replaced the standard Euclidean distance measure of Laplacian Eigenmaps with the Negative Geodesic Distance, a distance measure native to the multinomial simplex. We observe for the AMI subjectivity classification task that the resulting geodesic Laplacian Eigenmap reaches a first good operating point prior to the Euclidean variant, and for a spam detection variant that it performs significantly better in terms of averaged F1 and AUC scores. Moreover, for the second task we found that combining weights computed by the standard Euclidean Laplacian Eigenmap and its geodesic variant yields maximum performance. We interpret these results as evidence for the prevalence of hybrid geometry of L1-normalized data.

In Section 7.2, we pursued this line of research, and investigated whether noise elimination techniques can provide additional evidence for hybrid geometry. In that section, we showed that a geodesic variant of the Hein-Maier algorithm is capable of eliminating geodesic noise from Riemannian manifolds, and that the combination of geodesic and Euclidean noise elimination leads in all of the investigated tasks to significantly better performance of either one of the two single source noise elimination procedures. The performance of the combined noise elimination strategy was shown to be dependent on the variety in neighbor selection by the noise elimination algorithms. The more varied the merge of the two neighbor sets produced by the Euclidean and geodesic noise elimination algorithms, the higher the performance difference of the combined noise elimination strategy, as compared to the average single algorithm performance. We take the latter observations as evidence for the existence of hybrid geometry as well: if the geometry of the datasets were homogeneous (either Euclidean or geodesic), then combination of geodesic and Euclidean information would not be able to improve on either one of these data sources.

Concluding, then, these two techniques underline our hypothesis that L1-normalized document space should not be approached uniformly with geodesic distance measures. In Chapter 8, we will outline a procedure that allows us to determine the trade-off between Euclidean and geodesic distance measures, through a *back-off* classifier calibration strategy.

Chapter 8

Classifier calibration

In this chapter, we put to use our findings from Sections 7.1 and 7.2, where we have shown empirically that it is possible for L1-normalized document space to have hybrid geometry. In Section 7.1 we showed that hybrid geometry Laplacian Eigenmaps outperform single-geometry Eigenmaps for the purpose of feature selection, and in Section 7.2 we argued that a denoising algorithm based on both Euclidean and geodesic distance measures obtains the best results and outperforms single-geometry methods.

In Section 8.1, we first develop a calibration technique based on thresholding the decision function of a classifier. We demonstrate that this technique, applied to L1-normalized data, creates practical added value: it can be used to automatically determine the hard cases that cannot be labeled automatically with sufficient confidence by a classifier, given a pre-specified accuracy level. In Section 8.2, we subsequently generalize this technique: we demonstrate that by thresholding two distance measures (the geodesic and Euclidean distance metrics), we can learn how to separate our data in overtly geodesic and Euclidean parts, as well as a mixed, hybrid part. This procedure leads to increased classifier performance. We obtain additional evidence for our central thesis that L1-normalized document data may display hybrid geometry, and that entropy is the decisive demarcating principle.

8.1 Accuracy-based classifier calibration

From a user's perspective, classifier accuracy is usually perceived as an intuitive and easily conveyed evaluation criterion for classifier performance. In many practical application scenarios, disappointing accuracy results will hinder the application of classifiers. However, even weakly performing classifiers could in principle be successfully deployed in manual workflows: namely, when they are restricted to classifying the subset of cases they are known to perform well on, while meeting minimal quality requirements. An operationalization of this idea

would lead to a workflow according to which some work might be automated by a classifier, whereas the 'hard cases' that produce too much difficulty for the classifier would be delegated to humans. Such a workflow would call for a more articulate classifier evaluation criterion, where a classifier is tuned to optimal performance on the basis of *a priori* determined accuracy requirements. Its utility can then be computed as the amount by which it is able to realize these requirements. In this section (based on Kraaij et al. [2008]), we propose to augment a first-level classifier with a second-level or meta-classifier mapping all decisions of the first classifier that do not meet a pre-specified confidence value to a category 'for manual inspection'. The first-level classifier can be evaluated in terms of what we will call *yield*: the proportion of observations that can be classified automatically with a minimum pre-specified accuracy. Classifier yield can be viewed as an estimate of classifier deployability for a certain task.

8.1.1 Accuracy and yield

The intended use of the ensemble {*accuracy,yield*} is to measure the classifier yield at a fixed (minimum) level of accuracy. Table 8.1.1 shows a ternary contingency table with one additional label: "?" ('manual inspection'). On the basis of this contingency table, accuracy can be defined as usual:

$$accuracy = \frac{TP + TN}{TP + TN + FP + FN} \tag{8.1}$$

We subsequently define *yield*, the proportion of observations that is not labeled as M, as:

$$yield = \frac{TP + TN + FP + FN}{TP + TN + FP + FN + M} \tag{8.2}$$

Ground truth	Assigned		
	+	-	?
+	TP	FN	M
–	FP	TN	

Table 8.1: Ternary classification contingency table.

As far as we know, the proposed ensemble of measures (yield at minimum accuracy) is a novel way of measuring the quality of a classifier. There are several established extrinsic (task-based) traditions that have some elements in common, like *linear utility* (Hull and Robertson [1999]):

$$linear\ utility = \alpha TP + \beta FP + \gamma FN + \delta TN \tag{8.3}$$

and *detection cost* (Fiscus and Doddington [2002]; see Appendix B):

$$detection\ cost = C_{Miss} \times P_{Miss} \times P_T + C_{FA} P_{NT} P_{FA} \tag{8.4}$$

We refer the reader to Appendix B for a short overview. A commonly used evaluation procedure is to measure the false alarm rate at a fixed maximum false reject (miss) rate, or vice versa (Bolle et al. [2003]). Our proposed procedure is similar in the sense that a certain operating point can be pre-defined in order to compare systems. This pre-defined operating point provides an anchor in the recall-precision trade-off and simplifies evaluation to a single measure, just as the F-score defines a certain point in the precision-recall space. The implementation of these concepts in a classifier architecture is specified in Section 8.1.2.

8.1.2 Classifier setup

In our implementation of the two-level classifier architecture, the first level classifier consists of a Support Vector Machine using the NGD kernel. The SVM was modified to provide a posterior probability (a *Platt score*; Platt [2000]) as output in addition to the predicted class[1]. The Platt scores were subsequently used as input (σ below) for a *meta-classifier* which was implemented by a decision rule, based on two thresholds θ_l and θ_u:

$$Class(\sigma) = \begin{cases} + & if \quad \sigma => \theta_u \\ M & if \quad \theta_l < \sigma < \theta_u \\ - & if \quad \sigma <= \theta_l \end{cases} \tag{8.5}$$

Maximizing *yield* now boils down to minimizing M, the quantity of data that cannot be labeled with the pre-specified level of minimum accuracy. The thresholds maximizing the yield while satisfying the pre-specified minimum accuracy were computed through pseudo-exhaustive search by a two dimensional parameter sweep (for both threshold parameters θ_u and θ_l) on a development set. We used a small, fixed stepsize of 0.05. The development data set for parameter training should be chosen carefully, since we assume that the class distribution is the same in the development set and the test set and that the Platt score distribution is more or less similar in the development and test set, for both classes. The training process is outlined in Algorithm C.14.

8.1.3 Experiments

We evaluated the benefits of the two-level classifier architecture with two experiments. The first experiment concerns the detection of domestic violence in police files of the Regional Police Force Amsterdam-Amstelland (RPAA). The second experiment addresses the ECML 2006 spam detection data.

[1] In fact it is not essential for the classifier to output true probabilities, it can be any monotonously increasing ranking function.

Detection of domestic violence

The label *domestic violence* is not always correctly assigned to reports by the registrating police officer. It is therefore desirable to recognize these cases post-hoc automatically. Domestic violence has a complex definition where several conditions need to be checked. This makes the automatic recognition of domestic violence in police reports on the basis of textual cues a non-trivial task. Current practice for filtering out domestic violence cases from the full database of incident reports is based on a manually crafted rule-based system. The current rule set in use at RPAA generates a high number of false positives, necessitating a manual check. We compared two classifiers: a baseline rule-based classifier[2] using hand-crafted thesauri, and the ternary classifier discussed in Section 8.1.2. The rule-based classifier uses both lexical and phrasal features, such as 'my father beats'. The ternary classifier architecture uses the same lexical feature set as the baseline classifier, but does not use the phrasal features. Our main interest was to assess the effect of the ternary classifier on the reduction of manual checks, maintaining a required quality level. Training data consisted of a collection of 1,736 reports, manually re-checked, with 1,101 positive cases. A random sample of 200 case files was used for development, the rest (1,536) for training. Test data consisted of a collection of 2,291 reports, labeled by registrating officers, with 541 positive cases. The SVM and the Platt function were trained on the training data. Subsequently, optimal upper and lower decision score thresholds were computed using the development data with a pre-specified desired accuracy > 0.9.

	Accuracy
Baseline classifier	0.73
SVM	0.84

Table 8.2: Results for the detection of domestic violence on the full test set using a single classifier

Figure 8.1 shows the probability that the classifier is correct as a function of its Platt score. While the fit clearly is not optimal, the relation between posterior probability and Platt score is an increasing function.

Table 8.2 lists the evaluation results (measured in terms of accuracy) for the baseline rule-based ranking classifier and the SVM. The SVM has a superior performance compared to the rule-based system; however, its accuracy is still too low for actual deployment at RPAA. For the ternary classifier, however, after score thresholds were tuned on the development data, we were able to

[2]This classifier is actually a ranking system, where a decision threshold was chosen manually.

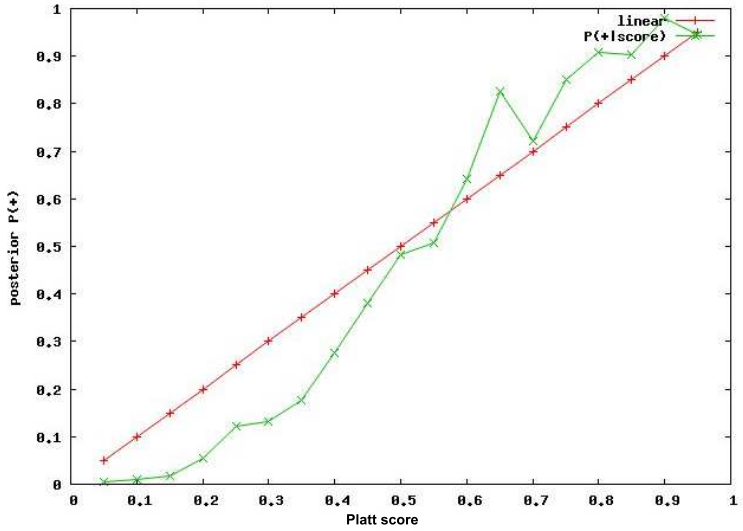


Figure 8.1: Posterior probability as a function of Platt score

automatically identify the reports where the classifier decision is based on a low confidence score, meeting a pre-specified accuracy level. We obtained a yield of 70% at an accuracy of 90%.

	Accuracy	Yield
Development set	0.90	0.70
Full test set	0.92	0.86
Test set sample A	0.93	0.86
Test set sample B	0.92	0.89
Test set sample C	0.93	0.86

Table 8.3: Results for the detection of domestic violence experiment using the ternary classifier.

Table 8.3 lists the accuracy and yield of the ternary classifier for development and test sets. As an additional diagnostic, three random samples of the test set (sample size = 1,000) were evaluated. The obtained accuracy and yield on the test set are both higher than on the development data. This could be explained by the fact that the test set was obtained from cases from a different year, where annotation standards might have changed. Still, results of the classifier on development and test set show the potential of the proposed

approach, which seeks to minimize the amount of human labeling while meeting pre-specified quality standards. The results at various subsamples demonstrate the robustness of the parameter settings. Related work on the same dataset explores the possibility of involving a human expert for an interactive selection and definition of complex features, based on formal concept analysis (Poelmans et al. [2008]).

Spam detection

For our second experiment we used the ECML 2006 Discovery Challenge data (see Section 7.1.3). We limited ourselves to task A, and divided the evaluation sets in a development set of 500 emails (used to estimate the thresholds) and an evaluation dataset consisting of the remaining 2,000 messages.

	Binary accuracy	Ternary accuracy	Ternary yield
User 1	0.62	0.89	0.19
User 2	0.65	0.90	0.39
User 3	0.78	0.91	0.69

Table 8.4: Results for the detection of spam emails using a binary and ternary classifier

Table 8.4 lists the results of the spam detection experiment. The first column lists the accuracy of the standard binary classifier (SVM with NGD kernel). The second and third column list the obtained accuracy and yield when the ternary classifier's thresholds have been set for a minimum accuracy level of 0.90 using the development subsets. The desired accuracy (0.9) can be achieved for about 20-70% of the email messages depending on the user, making this a much harder task than the domestic violence detection.

Figure 8.2 illustrates the optimal operation curves for each user in a *yield plot*, where the classifier yield is plotted as a function of the desired accuracy level. A yield plot can be used to decide visually on the desired trade-off between accuracy and yield, for a given task. For instance, it can be seen directly that for user 1, yield at an accuracy of 90% is only 20%, but that, when the accuracy constraint is relaxed to just under 80%, yield rises to around 60%.

8.1.4 Discussion and Conclusions

We have presented a new ensemble of evaluation measures for a situation where a classifier is used to partially replace human labelling effort. The measures accuracy and yield relate well to a more task-oriented or extrinsic view on evaluation, where the focus is on cost savings. Accuracy and yield can be seen as workflow-based measures for 'fidelity' and 'completeness'. The simplicity

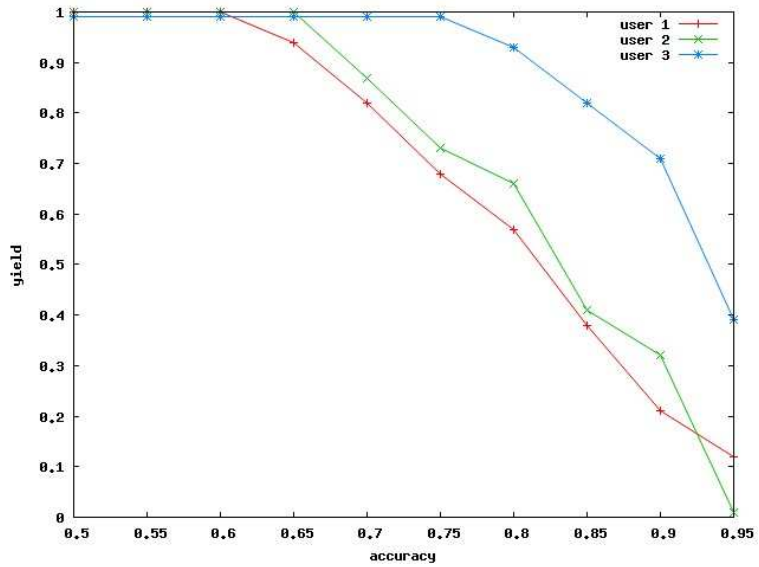


Figure 8.2: Yield as a function of (minimum) classifier accuracy, in the ternary classifier setting

of our approach does have some shortcomings. Accuracy as an aggregated measure hides the different sources of classification quality. It is well known that accuracy is sensitive to class imbalance. An alternative ensemble based on false alarm rate, false reject rate and yield would solve this problem. However, this ensemble might be less intuitive for non-experts.

A second contribution of this section is the concept of a ternary classifier forwarding cases that cannot be classified with a pre-specified accuracy to a human expert, thereby reducing the error rate of the classifier. We demonstrated the effect of the classifier on two real-life datasets, demonstrating its possibilities for factoring out hard cases from easy to classify cases with a high pre-specified accuracy.

8.2 Distance metric-based classifier calibration

In this section, we show that the calibration technique of Section 8.1 can be generalized to calibrate a classifier using the geometry of the data it is trained on. Given the assumption that a given dataset is best described by a hybrid Euclidean/geodesic geometry, we would like to identify the circumstances under which we can delegate our data to a Euclidean classifier, and when we should

deploy a geodesic classifier. As explained in Chapter 6, application of the NGD kernel is particularly effective for low-entropy, L1-normalized data.

The aim of this chapter is to gather evidence for the usefulness of entropy-based thresholding using a prototypical classifier. We expand the hybrid Euclidean/geodesic perspective on document space we initiated in the previous chapters, where the idea is that L1-normalized document representations allow for a hybrid representation as both a curved manifold and a Euclidean, flat manifold. The calibration technique we develop allows us to identify the operating point at which we switch from one manifold to the other.

8.2.1 Calibration procedure

The prototypical classifier architecture we will use in our experiments consists of a composite nearest neighbor classifier: a classification model that employs three types of distance functions: Euclidean, geodesic, and hybrid. The motivation for the k-nearest neighbor architecture stems from a desire for simplicity and generality (see the description of memory-based learning in Section 2.1). In order to determine which parts of the data are delegated to which classification strategy (Euclidean, geodesic, or hybrid), we will outline a procedure to threshold the classification model using the entropy of local nearest neighborhood sets. The procedure is as follows.

Given a training dataset, we first estimate two SVM models: one based on the NGD (geodesic) kernel, and one using a Euclidean distance metric (the linear dot product)[3]. Subsequently, for every element in a test dataset, we find its geodesic nearest neighbors in the reduced training dataset constituted by the SVM model based on the NGD kernel, and, likewise, its Euclidean neighbors using the linear kernel. Next, we calibrate the classifier by determining two thresholds θ_l and θ_u that demarcate the entropy of two subspaces: θ_l is an upper bound for the geodesic data, that –as we motivated in Section 6.1– displays relatively low entropy. The threshold θ_u on the other hand is the lower bound for datapoints best catered for by the Euclidean distance metric, which is most effective for high entropy data. First, define the *average vector entropy* of the nearest neighborhood of data point x as

$$\overline{H}(NN(x)) = \frac{1}{\mid NN(x) \mid} \sum_{i=1}^{|NN(x)|} -\sum_{j=1}^{d} y_{ij} \log_2 y_{ij} \tag{8.6}$$

where y_{ij} denotes the j-th component (feature) of the i-th vector $y \in NN(x)$.

For a given datapoint x, a relatively low-entropy geodesic neighborhood[4] with average entropy $\overline{H}(NN(x)) \leq \theta_l$ is essentially geodesic. In such a case, we will use the geodesic neighbors for voting for the class prediction of x. On

[3] We used the default value of the C hyperparameter for both kernels.
[4] A neighborhood determined by the geodesic distance function.

the other hand, any such neighborhood with relatively high average entropy $\overline{H}(NN(x)) \geq \theta_u$ is essentially Euclidean, and therefore useless from a geodesic point of view. In such a case, we will back off to the Euclidean neighborhood, using only the Euclidean neighbors for class prediction. In the case of relatively moderate entropy, we combine the Euclidean and geodesic neighborhood sets for predicting the class of x by adding the respective votes. Votes for classes are weighted. Every neighbor n with class c in the nearest neighborhood set for a test point x brings out a distance-weighted vote for c with quantity q (cf. Shepard [1987]).

$$q = e^{-\gamma\delta(x,y)} \tag{8.7}$$

In order to make use of the negative geodesic distance metric as a weight, we normalize it to values between 0 and 1. An optimal choice for the weighting parameter γ can be found by performing a grid search on development data, running this algorithm and varying γ until an optimum is found. In our case, we used the development data for estimating thresholds. In most cases, thresholds estimated directly from the test data were similar to thresholds estimated from the development data. Algorithm C.15 describes the process of obtaining the thresholds θ_l and θ_u from development data.

8.2.2 Experiments

We carried out the following experiments. First, for the sequential datasets DIMIN, GPLURAL, PP, CHUNK and NER, we compared the effect of thresholding on the position-aware and position-unaware pullbacks. Second, we applied the thresholding procedure to both the L1-normalized version of the AMI subjectivity dataset (Section 6.3), and the position-unaware pullback applied to this dataset[5].

For all experiments, data was uniformly split into 1/3 development and 2/3 training+test partitions; the training+test and development partitions were subsequently split into 1/3 test and 2/3 training partitions. We fixed as many as experimental parameters as possible: we uniformly set γ to 0.9, and we limited the nearest neighbor set size to 7; it is not our intention to obtain optimal classifier performance, but rather to compare the effects of thresholding on different datasets using a uniform classifier architecture.

Having a prespecified accuracy (or a quality measure such as F-score) to optimize for is not appropriate here: we want our thresholds to be optimally chosen such as to produce the highest accuracy or F-score as possible, and we have no specific interest in the size of the mixed neighborhood. Notice that it makes no difference to either favor the Euclidean neighbor sets or the geodesic neighbor sets for thresholding, as the thresholds are mutually exclusive, and the process of thresholding converges to the same thresholds in either case.

[5] As this data is essentially non-sequential, the position-unaware pullback would make no sense here.

8.2.3 Results

Results for the sequential tasks are listed in Table 8.5. These results should

Position-unaware pullback						
	Euclidean	geodesic	calibrated	%euclid	%geo	%hybrid
DIMIN	92	92.6	**94.8**	4	11	85
GPLURAL	85.6	84.9	**86.2**	7	3	90
PP	64.8	77	**79.9**	20.5	43.5	36
CHUNK (10K)	84.2	85.8	**86**	4	78	18
NER (10K)	41	45.7	**45.8**	0	88.2	11.8
Position-aware pullback						
	Euclidean	geodesic	calibrated	%euclid	%geo	%hybrid
DIMIN	96.8	97.2	**97.4**	11.6	88.4	0
GPLURAL	87.7	88	**88.9**	1.2	0	98.8
PP	65.6	69.5	**71.7**	19.5	63.2	17.3
CHUNK (10K)	89.9	**91**	**91**	7.3	0	92.7
NER (10K)	82.3	78.7	**83.4**	1	0	99

Table 8.5: Accuracy results (best scores in bold) and average geometric distribution (the distribution of the types of neighbors as used for classification of the test data) obtained with uncalibrated and calibrated classifiers for sequential data on fixed training/test splits.

be interpreted as follows. The accuracy scores for 'Euclidean' and 'geodesic' correspond to the scores an uncalibrated, fully Euclidean or geodesic k-nearest neighbor classifier ($k = 7$) would obtain. The 'calibrated' score corresponds to a calibrated, hybrid Euclidean-geodesic classifier. As such, these scores cannot be readily compared to the scores we obtained in Section 5.2, which were based on support vector machines, nor to the results in Daelemans and van den Bosch [2005], which were obtained with k-nearest *distance* classifiers involving a more advanced set-up (such as feature weighting, and the MVDM transform (see Section 5.4)). Our results do however outline the effects of classifier calibration on accuracy, and show that in all experiments carefully thresholded hybrid classifiers attain the best results. The percentages in the table are the average percentages of Euclidean ('Euclid'), geodesic ('geo') and hybrid Euclidean-geodesic neighbors used during the classification of the test data.

Notice the relatively large proportion of Euclidean neighbors for PP; this is in agreement with our previous findings for this dataset. For both pullbacks, calibration is able to boost results. For NER, the situation is reversed: the calibration procedure selects a predominantly large proportion of geodesic neighbors for this dataset; this task does not benefit much from the Euclidean neighbors under the position-unaware pullback; the fact that the position-aware

pullback achieves much better results underlines the essentially sequential organization of this task. Under the position-aware pullback, NER also appears to be an essentially mixed task: 99% of the neighbor types fall into the hybrid class, and within this class, neighbors are per definition dually represented as both Euclidean and geodesic.

The position-aware pullback leads in general to an increase of the mixed area between the two thresholds, at the expense of the number of geodesic neighbors compared to the position-unaware pullback: the percentage of mixed neighbors is increased from an average of 48.2% for the position-unaware pullback to an average of 61.6% for the position-aware pullback. Likewise, the average percentage of geodesic neighbors drops from 44.7% to 30%, and the average percentage of Euclidean neighbors increases minimally from 7% to 8.1%. The exceptions here are DIMIN, where the mixed category is reduced to zero, and the geodesic neighbors see an increase of 77.4%, and PP, where the amount of Euclidean neighbors remains the same, but the geodesic neighbors receive an influx of 20% from the mixed neighbors. In the latter case, this increase in geodesic neighbors appears to reduce performance considerably. When we compare the average gain in accuracy of both pullbacks, we see no big difference: on average, the mixed classifier outperforms the best single source classifier for the position-unaware pullback with 1.2%, and with .9% for the position-aware pullback (which attains much higher accuracy on average).

8.2.4 Non-sequential data

Next, we applied the thresholding algorithm to L1-normalized, essentially non-sequential data: the AMI sentiment dataset (Section 6.3). We used three meetings as development data, and applied 10-fold cross-validation to the remaining 10 meetings, using one meeting in turn as test data. We optimized the threshold finder for macro-averaged F-score, due to the class imbalanced nature of this data (107,723 negative cases, and 56,628 positive cases). The results in Table 8.6 and Figure 8.3 show that for the AMI sentiment data, thresholding on standard L1-normalized data does not significantly increase F-score (+1.3%). The data is of low entropy, and therefore the thresholding algorithm puts the majority of datapoints (69.6%) in the geodesic bin, and only 3.9% in the Euclidean bin. It is instructive to take a more detailed look at the AMI data. Many utterances in this dataset are relatively short, often consisting of just a few words that often have a frequency of 1 (they are, in the context of the utterance they occur in, *hapaxes*). Yet, these rather uniform frequencies do not automatically produce high entropy data in an absolute sense.

Average F_{macro}			
	Euclidean	geodesic	calibrated
L1	50.9	54.7	56
Pullback	51.1	57.5	**61.3**
Average geometric distribution			
	% Euclidean	% geodesic	% hybrid
L1	3.9	69.6	26.5
Pullback	44	46.7	9.3

Table 8.6: Macro-averaged F-scores and geometric distribution (averaged percentages) obtained with uncalibrated and calibrated classifiers for the AMI data (13-CV).

The breakdown of the data in terms of utterance length is given in Figure 8.4. It shows that most utterances are very short on average: 82.4% have a length of maximally 10 words (31.2% of length 1, and 51.2% with a length of 2-10 words).

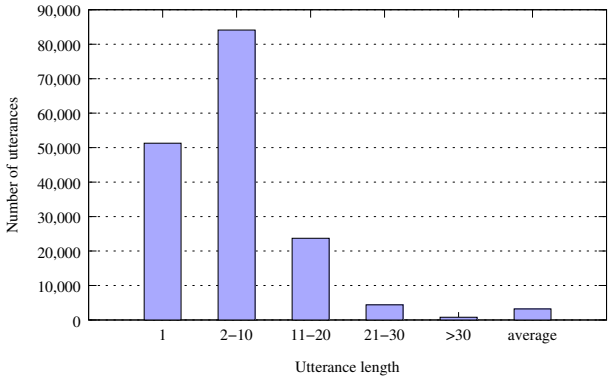


Figure 8.4: Break-down of AMI sentiment data in terms of utterance length.

The relation between utterance length and entropy is illustrated by Figure 8.5: assuming all words are hapaxes (i.e. all words in a certain utterance occur just once in that specific utterance), entropy as a function of utterance length clearly is a logarithmic function. It would seem that this relationship between utterance length and entropy leads to undesired effects: long utterances automatically produce higher entropy data, and shorter utterances, even when they are entirely made up of words with a frequency of 1, will produce low entropy data.

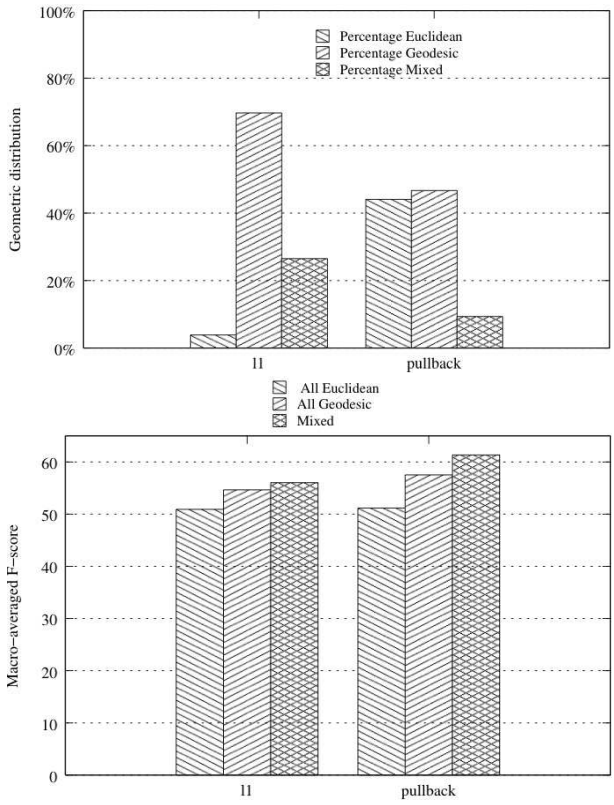


Figure 8.3: Average macro-averaged F-score (over 13 folds) for the AMI data, for the homogeneous Euclidean and geodesic classifiers, and the thresholded, mixed classifier.

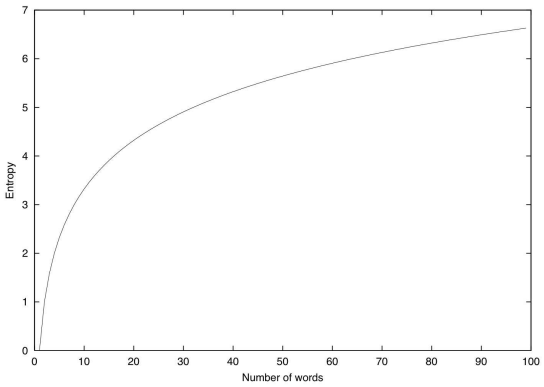


Figure 8.5: Entropy as a function of utterance length.

This situation changes when we apply the position-unaware pullback to the data. For this type of non-ordered data, the pullback produces an ordered sequence of probabilities for a document, where every i-th probability $p_i(w \mid C)$ is the probability of observing the i-th word w in conjunction with class C. The pullback appears to trigger an increase in entropy: the average increase in entropy for the 13 folds of the AMI data is 59%. This is quite logical: the pullback effectively ignores frequencies inside a bag of words, and, instead, derives count information based on joint occurrences of features and classes from *outside* the bag of words. This smoothing process will, in a number of cases, lead to increased entropy: namely, those cases where frequencies of words vary wildly across a certain document, but the co-occurrences of features and classes are similar across classes. Apart from this, the pullback increases the number of features as well: up to $C \cdot N$ for C classes and N features, which, in the case of a flat probability distribution, only serves to increase entropy, as we saw in Figure 8.5.

Results first of all show that the pullback leads to significantly higher macro-averaged F-score ($p < .01$ with the non-parametric Wilcoxon test). Second, we see from the distribution of neighbors (geodesic, Euclidean or mixed), as produced by the thresholding, that the pullback generates many more Euclidean neighbors. This is in agreement with the increase of entropy, and thresholding proves to be particularly beneficiary.

The calibration technique we outlined serves to focus the geodesic distance measure on truly geodesic data. It demarcates the boundaries within which the geodesic distance measures yields optimal performance. Calibrating a classifier on the basis of entropy of local neighborhoods addresses the two risks we identified: the risk of dot products reaching 1 for strongly overlapping vectors, and the risk of dot products producing low values for highly dissimilar vectors. As noted in Section 6.2, these are the cases where the inverse cosine-based NGD kernel performs poorly. Factoring out these cases and handing them over to a Euclidean kernel turns out to be beneficiary.

8.2.5 Related work

Our work is related to recent work on local metric learning in manifolds, e.g. Abou-Moustafa and Ferrie [2008], who fit an ellipsoid neighborhood to test points using a parametrized version of the Mahalanobis distance. As these authors note, the Euclidean distance ignores the structure, scale, variance and correlations in the data. Lebanon [2006a] states that, in the absence of clear evidence of Euclidean geometry, metric structure should be inferred from the data. We therefore can view Euclidean approaches to document classification as back-off options as compared to the more intricate, structure-aware geodesic approaches. This is in the same spirit as the division of labor between more complex n-gram models and simple unigram models in language modeling which

is known as a back-off model (c.f. Katz [1987]). Along similar lines of reasoning, the proposed calibration technique can be interpreted as a *back-off* strategy for document geometry estimation: whenever we determine that the information in the curved manifold is unreliable for accurate classification, we switch to the dual representation of the data in terms of a Euclidean manifold, and perform classification based on Euclidean distances. Boundary cases are treated with a mixed classification strategy, combining information from both manifolds.

8.2.6 Kernelization

The thresholding approach can be kernelized, i.e. expressed as an operation on kernels. One can construct a hyperkernel consisting of three linearly interpolated subkernels. The subkernels correspond to the Euclidean, geodesic and mixed distance measures. The weights interpolating them are binary indicators based on the two thresholds on the entropy of the local neighborhoods, as follows. Given two data points x, y, with y in the local neighborhood of x:

$$\begin{aligned} K_{HYPER}(x,y;\theta_l;\theta_u;\gamma) = \quad & \lambda_1 K_{expo \circ NGD}(x,y;\gamma) + \\ & \lambda_2 K_{expo \circ NGD \circ EUCLID}(x,y;\gamma) + \\ & \lambda_3 K_{expo \circ EUCLID}(x,y;\gamma) \end{aligned} \tag{8.8}$$

The subkernels are defined as follows:

$$\begin{aligned} & K_{expo \circ NGD}(x,y;\gamma) = e^{-\gamma \delta_{NGD}(\mathbf{x},\mathbf{y})} \\ & K_{expo \circ EUCLID}(x,y;\gamma) = e^{-\gamma \delta_{EUCLID}(\mathbf{x},\mathbf{y})} \\ & K_{expo \circ NGD \circ EUCLID}(x,y;\gamma) = \frac{e^{-\gamma \delta_{NGD}(\mathbf{x},\mathbf{y})} + e^{-\gamma \delta_{EUCLID}(\mathbf{x},\mathbf{y})}}{2} \end{aligned} \tag{8.9}$$

and the weights are based on the thresholds θ_l, θ_u:

$$\begin{aligned} & \lambda_1 = 1 \quad if\ \overline{H}(NN(x)) \leq \theta_l; \quad 0\ else \\ & \lambda_2 = 1 \quad if\ \theta_l < \overline{H}(NN(x)) < \theta_u; \quad 0\ else \\ & \lambda_3 = 1 \quad if\ \overline{H}(NN(x)) \geq \theta_u; \quad 0\ else \end{aligned} \tag{8.10}$$

8.3 A haversine kernel

In Chapter 6, we surmised that the poor performance of geodesic distance measures on high entropy data was related to the well-known fact that the arccos function produces large rounding errors for small distances. Small distances are inherent to high entropy data, where dot products of vectors can easily produce small values. Further, non-linear behavior of the arccos functions for distances close to 1 typically arises for strongly overlapping maximum entropy vectors. In that chapter we also related entropy to distance: high entropy data is represented on a manifold by compact regions with small distances between points.

In the previous sections, we have demonstrated that high entropy data can be better delegated to a Euclidean kernel instead of the NGD kernel. We found that thresholding the geodesic distance function effectively amounts to thresholding the error component of the arccos function.

In this section, we investigate if there is any benefit in using alternative distance measures that do not suffer from inaccuracy for small distance data to begin with. We will propose such an alternative kernel: a kernel based on the so-called *haversine distance*. The haversine distance, defined on non-Euclidean manifolds, stems from cartography and is widely known for its accuracy for small distances (Sinnott [1984]). We compare a kernel based on the haversine distance with the NGD kernel on a number of document classification tasks, investigating its adequacy for dense, high entropy data.

8.3.1 The haversine kernel

The *versine* function, defined as $versine(x) = 1 - \cos(x)$, occurs historically in navigation as the haversine, or *half the versine*: $haversine(x) = \frac{1}{2}versine(x)$. Given two points A and B on a unit sphere, the haversine is defined on angle θ as illustrated by Figure 8.6. In general, for a sphere with radius R, the haversine distance can be computed from the haversine function as $2R \cdot \arcsin(haversine(\theta))$. It can be kernelized as follows:

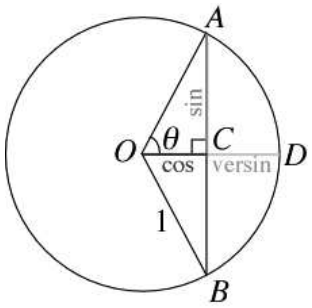


Figure 8.6: The versine on a unit circle.

$$K_{HAV}(\theta, \theta'; R) = 2R \cdot \arcsin\left(\sqrt{\frac{1 - \cos(\sum_i \theta_i \theta'_i)}{2}}\right) \tag{8.11}$$

This kernel has one hyperparameter, R, the radius of the sphere. Since we operate on the unit sphere (by L1-normalization), $R = 1$, and hence $2R = 2$.

Keeping R as a separate hyperparameter was observed to have clear effects in the experiments reported below, however, and we therefore tune $R \geq 1$.

The cos function is applied to the dot product of the two vectors θ and θ', which, when they are of unit length[6], is equal to the angle between them. In the case of L1-normalized data, vectors are not guaranteed to be unit vectors[7].

8.3.2 Positive definiteness

Prior to proving that the haversine kernel is positive definite, we mention the following (see Schölkopf and Smola [2002]):

8.3.1. PROPOSITION. *A kernel $K(\theta, \theta') = f(\langle \theta, \theta' \rangle)$ defined on the unit sphere in a Hilbert space of infinite dimensions is positive definite (pd) if and only if its Taylor series expansion has only nonnegative coefficients:*

$$f(t) = \sum_{n=0}^{\infty} a_n t^n \text{ with } a_n \geq 0 \tag{8.12}$$

Analogous to the line of reasoning in Zhang et al. [2005] when proving positive definiteness of the NGD kernel, we prove the following

8.3.2. PROPOSITION. *The kernel $K_{HAV}(\theta, \theta')$ on the multinomial manifold is positive definite.*

Proof The Maclaurin series (the Taylor expansion of a function about 0) for the arcsin function is

$$\arcsin(x) = \sum_{n=0}^{\infty} \frac{\Gamma(n + \frac{1}{2})}{\sqrt{\pi}(2n+1)n!} x^{2n+1} \tag{8.13}$$

with $\Gamma(x)$ the gamma function

$$\Gamma(z) = \int_0^1 \left[\ln \left(\frac{1}{t} \right) \right]^{z-1} dt. \tag{8.14}$$

The Maclaurin expansion for the cos function is

$$cos(x) = \sum_{n=0}^{\infty} \frac{(-1)^n}{(2n)!} x^{2n} \tag{8.15}$$

[6] A vector $\mathbf{v}$ is of unit length if its magnitude or norm, defined as $\|\mathbf{v} \cdot \mathbf{v}\| = \sqrt{\sum_i v_i^2}$ is 1. Any vector can be converted to unit length by dividing every component by the magnitude.

[7] In the experiments reported below, we did not find benefits from normalizing vectors to unit length.

Since we are on the multinomial manifold, where all data is scaled between 0 and 1, the dot product $\langle \theta, \theta' \rangle$ is always between 0 and 1 as well. This is due to the fact that $x \cdot y < x$ and $x \cdot y < y$, $\forall\ x, y \in \mathbb{R}$, which means that since $\sum_i^n x_i = 1$ and $\sum_i y_i = 1$, $0 < \sum_i x_i y_i \leq 1$. Since $cos(0) = 0$ and $cos(1) = 0.54$, the cos term will always be positive, and the argument of arcsin will be in $[\sqrt{0.23}, \sqrt{0.5}) = [0.5, 0.79]$. Let

$$f(x) = 2R \cdot \arcsin(x)\ \forall x \in [0.5, 0.79]. \tag{8.16}$$

We then set

$$K_{HAV}(\theta, \theta') = f(\langle \theta, \theta' \rangle) \tag{8.17}$$

and

$$f(x) = \sum_{n=0}^{\infty} c_n x^{2n+1} \tag{8.18}$$

where

$$c_n = \frac{2R\Gamma(n + \frac{1}{2})}{\sqrt{\pi}(2n+1)n!} \tag{8.19}$$

Since $\Gamma(x) > 0\ \forall x > 0$, and $R > 0$, we have $c_n > 0\ \forall n = 0, 1, 2, \ldots$. From Proposition 8.3.1, it follows that the dot product $f(\langle \theta, \theta' \rangle)$ is pd, which makes K_{HAV} pd. ■

8.3.3. Corollary. *The kernel $K_{HAV}(\theta, \theta')$ on the multinomial manifold is conditionally positive definite.*

Proof Every pd kernel is cpd (Schölkopf and Smola [2002]). ■

The haversine kernel is based on the cosine, rather than the inverse cosine of the NGD kernel. This means that, whenever the dot product between two vectors approaches 1, the cosine approaches 0.5, its maximum for this type of data. The arcsin function, taking values in the interval $[0.5, 0.79]$, produces distances in the range $[0.5, 0.9]$, a compressed interval compared to the NGD kernel, which produces distances in the range $[-\pi, 0]$. This implies that the haversine kernel represents distances on a more dense scale, for which it is known to yield good performance. For high entropy data, this interval is further compressed to an even narrower interval, with values approximating 0.79. Our hypothesis is that the haversine kernel outperforms the NGD kernel on this type of data, with its abundance of small scale distances.

8.3.3 Experiments and results

We performed four tests with the haversine and NGD kernels in order to validate our hypothesis that the haversine kernel outperforms the NGD kernel on dense, high entropy data. First, we compared the two kernels on the ECML 2006

Task A	NGD	haversine
Accuracy	67.6	**75.7**
Recall	**87.5**	79
Precision	62.8	**74.4**
F_1	73.1	**76.6**
AUC	**82.9**	81.6

Task B	NGD	haversine
Accuracy	61.5	**66.9***
Recall	75.9	72.7
Precision	59.3	**65.3***
F_1	66.4	**68.6****
AUC	61.5	**74***

Table 8.7: Results for the ECML 2006 Spam Detection task. Results in bold indicate better performance, and an asterisk indicates significant improvement. Significance results are only reported for Task B (15 observations), as Task A consists of just three observations. All significance results were measured with a paired t-test at $p < .01$ except the **-case, where $p < .1$, and which indicates a trend rather than a significant result.

spam detection task (see Section 7.1.3). Second, we addressed the problem of prepositional phrase (PP) attachment (Chapter 5.2). Subsequently, we applied the kernels to subjectivity classification of the AMI subjectivity data (Chapter 5). Finally, we measured the performance of the haversine kernel on random data with varying entropy, in order to measure the effect of entropy on classifier performance. In our experiments, no special effort was made to produce optimal feature sets or representations. Instead, our aim is to compare the NGD kernel with the haversine kernel across a number of tasks. Therefore, we are interested in relative performance rather than optimal performance. As mentioned in Section 2.5.2, the NGD kernel is hyperparameter-free, and Zhang et al. [2005] found no use in tuning the general cost hyperparameter C for this kernel (see Section 2.1). Following Zhang et al. [2005], who did not observe any benefits for optimizing C for the NGD kernel, we did not optimize this hyperparameter for NGD either, but we did optimize it for the haversine kernel using heldout development data, together with the R hyperparameter.

8.3.4 ECML 2006 Spam Detection Task

As our first experiment, we chose the ECML spam detection task introduced in Section 7.1.3. The results for the ECML data, presented in Table 8.7, put us on an average position compared to the official results reported for this challenge: for Task A, average AUC is 82.9 for all participants, and for Task B the average AUC is 74.6[8]. The area *above* the ROC curve (AAC, defined as 1-AUC%) of the two kernels for the two tasks (17.1/18.4 for task A, and 38.5/26 for task B) can be compared to e.g. the ECML 2006 Task B winning solution of Cormack [2006], who reports AAC scores of 7 and 24.8 for an SVM

[8]Ranked results are obtainable from the ECML 2006 Discovery Challenge website.

PP	NGD	haversine
Accuracy	**82.9**	80.3
Recall	**86.8**	79.9
Precision	**75.3**	74.1
F_1	**80.6**	76.9
AUC	**91.2**	72.4

Table 8.8: Results for the PP dataset.

optimized with Dynamic Markov Modeling. While this by itself is an interesting result (we made no attempt to define informative features, no feature selection was attempted, and all features consisted just of normalized word frequencies), more important is the observation that the haversine kernel outperforms the NGD kernel on both tasks, producing (significantly) higher accuracy, precision and F-scores.

8.3.5 PP attachment

Recall that the PP attachment task put forward by Ratnaparkhi [1998a] and introduced in Section 5.2, consists of deciding for verbal or nominal attachment of a PP on the basis of a four-cell word-based window that contains a main verb, a noun, a preposition, and a noun. We converted this data to L1-format, which in general produces frequencies of 0.25, since every word occurs often just once. Since the entropy of a probability distribution is logarithmic in the number of variables, the short feature vector size of this data (4) leads, in terms of bits, to a quite uniform low average vector entropy of 1.9 bits. Yet, the maximum entropy representation of this type of data would produce an average vector entropy of just 2 bits. In this experiment, we used the standard training (20,801 instances) and test (3,097 instances) split of this data (Ratnaparkhi [1998a]). The accompanying development data (4,039 instances) was used for hyperparameter tuning.

As it turns out (see Table 8.8), the NGD kernel outperforms the haversine kernel for this task, yielding consistently higher scores for recall and accuracy. This can be explained by the fact that there is very little overlap between the various feature vectors in this data, which means that the vector product that is part of the NGD kernel uniformly produces values far below 1. So, maximum entropy data increases the chance of dot products to approach 1, but the overlap between feature vectors effectively is the most important factor for this to happen. Apparently, the dot product values are also not very small, which would have led the NGD kernel to produce rounding errors.

8.3.6 AMI subjectivity

For the AMI subjectivity classification dataset, we used character trigrams as feature representations. Three meetings were used as development data for hyperparameter estimation. For two specific meetings, we measured the performance of both kernels. The meetings were selected on the basis of their vector entropy[9] distributions, under the hypothesis that high entropy data will set back the NGD kernel, and favor the haversine kernel.

We performed two experiments: one (condition 'A' in Table 8.9) uses all training data, and the other (condition 'H') uses and classifies only the data that has vector entropy higher than average. Results obtained by the two kernels on these two meetings are listed in Table 8.9. Using all training data shows that the NGD kernel outperforms the haversine kernel, for both meetings. However, after selection of the high entropy data for the first meeting, the haversine kernel applied to this high entropy data outperforms the NGD kernel. For the second meeting, with its even spread of entropy data, this effect predictably does not occur, since most entropy values are around the average value.

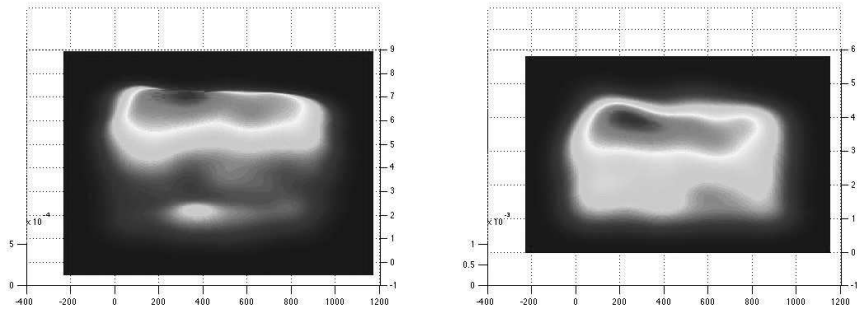


Figure 8.7: Vector density estimation for two AMI meetings (subjectivity data). The left picture displays the density for the high entropy data of Meeting 1, and the right picture for Meeting 2.

In order to visualize the situation, Figure 8.7 displays the result of a bivariate Gaussian parametric density estimation of the two distributions (Botev [2006]) under the condition 'H'. For all instances in the test data, the vector entropy is plotted on the y-axis. Light-colored areas represent high density. Clearly, the first meeting training data displays high-density entropy in the higher regions only, with entropy values much higher than for the second meeting. For the second meeting, entropy values are spread even across the spectrum, and values are relatively low.

[9] See Eqn. 8.6.

Meeting 1 (A)	NGD	HAV
Accuracy	**52.5**	49.1
Recall	21.9	**24.3**
Precision	**46**	40.6
F_1	29.7	**30.5**
AUC	**59.7**	50.7

Meeting 2 (A)	NGD	HAV
Accuracy	**70.9**	66.4
Recall	**51.3**	46.4
Precision	**71.7**	64.2
F_1	**59.8**	53.9
AUC	**71.5**	68.7

Meeting 1 (H)	NGD	HAV
Accuracy	45.8	**47.1**
Recall	31.8	**42**
Precision	**61.4**	59.8
F_1	41.9	**49.3**
AUC	42.9	**47.6**

Meeting 2 (H)	NGD	HAV
Accuracy	**67.3**	60.5
Recall	**62.4**	56.9
Precision	**72.8**	65
F_1	**67.2**	60.7
AUC	71.7	**71.9**

Table 8.9: Results for AMI subjectivity data (best results in bold). Condition A uses all training data, and condition H only uses training data that has vector entropy higher than average.

8.3.7 Random data

In order to closely investigate the relationship between entropy and classifier performance, we next generated 10 random, synthetic datasets consisting of the ground truth of the Task A Spam training data (2,500 cases), and a variable number of features (20-200, varied in steps of 20), with random frequency counts. The random counts were L1-normalized. The entropy range we obtained is from 4.1 (20 features) to 7.4 bits (200 features). We trained the haversine kernel on the first 2,000 items of each dataset, and tested it on the last 500 items, where hyperparameters were estimated on heldout data. The resulting F_1-scores were paired with the entropy values computed for every dataset, and were analyzed by the non-parametric Spearman test. This test produced a ρ value of 0.69, which indicates a strong positive correlation between entropy and classifier performance: the higher the entropy, the better the resulting classifier performance. This is in line with our previous remarks on the applicability of haversine-based distance measures to high entropy data. In Section 6, we measured for the NGD kernel an opposite trend: the higher the entropy, the lower the classifier performance.

8.3.8 Conclusions

In this subsection, we have proposed a novel kernel based on the haversine distance. The haversine distance is known to be a more accurate estimator for small distances than geodesic distance measures based on the inverse cosine, which are known to suffer from relatively large rounding errors for short distances. We proved positive definiteness of the haversine kernel, and demonstrated its compression effect on data. We subsequently showed its applicability to dense, high entropy data, which, as we argued, produces the necessary precondition of small distances. The results of the experiments support our hypothesis that the haversine kernel is applicable to dense, high entropy data, just like the Euclidean kernel. For the high entropy ECML spam data, the haversine kernel outperforms the NGD kernel, as well as for the dense, high entropy part of the AMI subjectivity data. The existence of these dense, high entropy regions of a dataset can be verified using e.g. a parametric density estimator, applied to the vector entropy scores of the instances in the data. On the PP attachment data, the NGD kernel outperforms the haversine kernel. The maximum entropy representation of this data does not automatically produce ultrashort distances: the maximum entropy representation theoretically leads to dot products approaching 1, which are in the non-linear part of the arccos function (see Figure 6.2). This happens for strongly overlapping vectors. For other vectors, the maximum entropy representation produces small values, which, when small enough, trigger rounding errors of arccos. This apparently does not occur in this data. The results on the synthetic data reveal a strong positive correlation between classifier performance and entropy: the higher the entropy, the better the performance of the haversine kernel.

For future work, we are interested in developing a division of labor between the NGD kernel and the haversine kernel similar to our approach for calibrating the NGD and Euclidean kernels, as both the NGD and haversine kernel appear to target different substrata of data represented on the multinomial manifold.

8.4 Summary

In this chapter, we presented a classifier calibration technique that is able to calibrate a classifier for a pre-specified, given accuracy, factoring out hard cases for manual inspection. We subsequently generalized this technique to a thresholding procedure that calibrates a classifier on the basis of distance metrics. We demonstrated for a number of sequential tasks that this approach is beneficiary, and increases accuracy. For non-sequential data, the pullback transform we put forward in Chapter 5 generates the prerequisites for this thresholding approach: it effectively increases entropy, producing data amenable to thresholding. The result is a significant increase of classification quality, as measured with macro-averaged F-scores. We conclude from these results that distance-metric based

thresholding is a viable technique, and the results support our central thesis: the natural geometry of L1-normalized textual data is neither purely geodesic nor Euclidean, but rather a mixture of both, depending on the entropy of local neighborhoods. We subsequently demonstrated the use of the haversine kernel: a novel kernel based on a distance measure that is known to be quite accurate for small distances. We proved the positive definiteness of this kernel, and demonstrated its applicability to dense, high entropy data.

Chapter 9

Conclusions

In this chapter, we summarize the findings of our work by formulating answers to the research questions raised in Section 1.1, and describe our ideas for future work. The problem statement of this thesis, as formulated in Section 1.1, was the following:

1. Can we extend the standard multinomial language learning apparatus to heterogeneous data, and sequential, limited context classifications tasks? (Research questions 1 and 2)

2. Can we motivate the existence of hybrid geometry in L1-normalized document representations? (Research questions 3 and 4)

3. If such hybrid geometry can indeed be motivated, how can we calibrate classifiers operating on this document space such that their performance is optimized? (Research question 5)

9.1 The Research Questions

In the next subsections, we formulate our answers to the research questions presented in Section 1.1.

9.1.1 Research question 1: heterogeneous information

The multinomial framework presupposes L1-normalized frequency data. The process of L1-normalization, working on bag representations of strings (features), does not discriminate between different sources of information: all strings in the bag are treated on a par, irrespective of eventual subsumption relations or different informativeness.

In order to accommodate the combination of heterogeneous information in one classifier, we proposed the concept of *hyperkernels*, which implements the

well-known idea of linear interpolation from the field of generative language modeling. A hyperkernel consists of a number of subkernels that are combined in a weighted manner. This weighting is a form of interpolation; hence, the hyperkernel embodies a form of kernel interpolation.

We were able to demonstrate that kernel interpolation corresponds to a pullback on the Fisher information, and implements Tikhonov regularization of submanifolds. We empirically evaluated the benefits of kernel interpolation for word unigrams and bigrams on polarity classification (sentiment analysis) of movie reviews. Our results rank this approach among the best performing on this data, and show that discrimination between different sources of information within the multinomial framework is feasible, theoretically well-motivated, and empirically desirable.

9.1.2 Research question 2: sequential and limited context tasks

The multinomial approach to text classification assumes data with a bag of words structure: unordered collections of words with repetitions, making up for discriminative frequencies. A large class of language learning tasks does not meet this constraint, however: for instance, many analysis tasks are windowed, feature-based tasks, such as part-of-speech tagging on the basis of a fixed window containing parts of speech of the left context of a target word, and the word forms of the right context. Frequency counts would not make much sense here, as most items occur once (e.g. a feature 'part of speech of the previous word').

In order to extend the multinomial classification framework to sequential or limited context tasks, we defined two data transforms

- A position-unaware transform, that counts co-occurrences of classes and features, irrespective of the position of each feature in the feature vector.

- A position-aware transform, that counts co-occurrences of classes and features, taking into account the exact position of a feature in a feature vector.

These class-feature co-occurrence transforms allow us to escape from the data bottleneck inherent to the limited information available for sequential or limited context tasks: we now use global co-occurrence information about joint manifestations of features and classes from the entire training data. We demonstrated that the data transforms correspond to pullbacks of the Fisher information, and produce competitive results on a number of sequential classification tasks. In addition, we proposed an interpolation strategy, allowing for backing-off from e.g. a bigram model to a unigram model. This interpolation strategy can be

used to weigh different sources of information. It establishes a direct relationshiop between multinomial classification and generative models of classification, such as language models, where this type of interpolation is quite common.

9.1.3 Research question 3: high entropy data and geodesic distance

A formal and empirical analysis of the Euclidean and geodesic distance metrics revealed that they become isometric in the case of maximum entropy data, which, as expected, displays maximum density (and hence ultrashort distances between datapoints). Our analysis is based on the entropy of vectors, which, under the multinomial framework, are probability distributions. We have demonstrated that the use of the inverse cosine inherent to the geodesic distance measures leads to undesired side effects on the two extrema of its domain: for small values leading to long distances, large rounding errors occur, and for values close to 1, leading to short distances, non-linear behavior of the inverse cosine becomes quite dominant, with steep descents in its output values. The latter phenomenon produces unjustified large differences between high-entropy vectors that are actually quite similar. Maximum or high entropy data contributes to both of these phenomena: we demonstrated that maximum entropy data produces the lowest scores (and hence the longest distances) possible for dissimilar vectors, and for similar vectors, maximum or high entropy frequencies produce values close to 1 (and hence the closest distances).

9.1.4 Research question 4: hybrid geometry

On the basis of two suites of experiments (dimensionality reduction and manifold denoising), we provided empirical evidence for the hypothesis that L1-normalized documents should not be uniformly analyzed with geodesic distance measures, but, depending on the entropy structure of the data, with either Euclidean, geodesic, or hybrid distance measures, combining Euclidean and geodesic distances. Our contributions are the following.

We proposed a dimensionality reduction approach based on geodesic and hybrid (combined Euclidean/geodesic) Laplacian Eigenmaps. The geodesic feature selector outperforms the Euclidean feature selector on the two datasets used. For one dataset, we obtained a significant performance gain by using the hybrid feature selector, which demonstrates that geometry is data dependent –and not so much: data *representation*-dependent.

Subsequently, a manifold denoising method based on Euclidean and geodesic distance measures also provided evidence for hybrid document geometry. Manifolds can be cluttered by noisy data. For noise elimination methods to operate optimally, awareness of the geometry of the manifold is important. We studied a recently proposed geometry-aware noise elimination algorithm based on

the Euclidean distance measure, and created a variant based on geodesic distance, as well as a hybrid variant. After creating an evaluation strategy using a transductive classifier, we showed in a number of experiments the benefits of averaging information from the Euclidean and geodesic noise elimination algorithms. Together, these experiments support our hypothesis that L1-normalized data should not be uniformly approached with geodesic distance measures, but that Euclidean distance measures may under certain conditions be preferable. These conditions, as we outlined, are related to the entropy of local neighborhoods used during the classification process.

9.1.5 Research question 5: classifier calibration

After having established empirical evidence for the possibility of hybrid geometry of L1- normalized data (or, put differently: evidence against a uniform treatment of L1-normalized data with geodesic distance measures), in order to determine the exact trade-off between Euclidean and geodesic distance measures, we proposed a classifier calibration technique.

First, we demonstrated that a carefully thresholded meta-classifier is able to factor out hard cases in a real-world workflow, handing them over to manual inspection. This thresholding procedure was subsequently generalized to a general classifier calibration approach. Under this approach, documents are represented dually by both a Euclidean and a curved manifold, and, based on thresholds on the entropy of the local neighborhood of a test point, the classifier uses either information from one of the manifolds, or a mixture of both. We argued that this is an instance of locally adaptive metric learning, and demonstrated the benefits of entropy-based classifier calibration on a number of data sets.

In addition, we proposed a novel kernel based on the haversine distance, a distance measure that is particularly accurate for ultrashort distances. We proved positive definiteness of this kernel, and applied it to two datasets. Our experiments demonstrate the usefulness of this kernel to dense, high entropy data.

9.2 Future work

A logical continuation of our work is the implementation of the calibration mechanism outlined in Chapter 8 in full-fledged classifiers, such as Support Vector Machines. In order to do so, we need to connect the idea of local neighborhood we used for our prototypical k-NN classifiers to the support vectors in Support Vector Machines. Our conjecture is that the density of the support vectors on the margins of the hyperplane can be used to do so. Apart from this, distance measures that display less distortion on ultrashort distances akin to the haversine distance measure may lead to interesting new kernels especially suited for

mixed and high entropy textual data. The exact trade-off between these types of kernels and the standard geodesic kernels is still to be determined. Finally, our sketch in Appendix D of the connection between document representations and quantum information science may contribute to interesting new, massively parallel algorithms for document classification.

Bibliography

K. Abou-Moustafa and F. Ferrie. Local metric learning on manifolds with application to query-based operations. In *Proceedings of the 7th Int. Workshop on Statistical Pattern Recognition (S+SSPR)*, pages 862–873, 2008.

M. H. Aghdam, N. Ghasem-Aghaee, and M. E. Basiri. Text feature selection using ant colony optimization. *Expert Syst. Appl.*, 36(3):6843–6853, 2009.

E. Aïmeur, G.Brassard, and S. Gambs. Machine learning in a quantum world. In *Proceedings of the 9th Conference of the Canadian Society for Computational Studies of Intelligence (Canadian AI 2006)*, pages 431–442, 2006.

E. Aïmeur, G. Brassard, and S. Gambs. Quantum clustering algorithms. In Zoubin Ghahramani, editor, *Proceedings of the 24th International Conference on Machine Learning (ICML'07)*, volume 227, pages 1–8, 2007.

A. Andreevskaia, S. Bergler, and M. Urseanu. All blogs are not made equal, exploring genre differences in sentiment tagging of blogs. In *Proceedings of the International Conference on Weblogs and Social Media (ICWSM-2007)*, pages 195–198, 2007.

R. Beals, H. Buhrman, R. Cleve, M. Mosca, and R. de Wolf. Quantum lower bounds by polynomials. In *Proceedings of the 39th Annual IEEE Symposium on Foundations of Computer Science (FOCS 1998)*, pages 352–361, 1998.

M. Belkin and P. Niyogi. Laplacian eigenmaps and spectral techniques for embedding and clustering. In *Advances in Neural Information Processing Systems 14*, pages 585–591. MIT Press, 2002.

M. Belkin, P. Niyogi, and V. Sindhwani. Manifold regularization: a geometric framework for learning from labeled and unlabeled examples. *Journal of Machine Learning Research*, 7:2399–2434, 2006.

F. Benamara, C. Cesarano, and D. Reforgiato. Sentiment analysis: adjectives and adverbs are better than adjectives alone. In *Proceedings of the International Conference on Weblogs and Social Media (ICWSM-2007) Boulder, CO, March 26-28*, pages 203–206, 2007.

Y. Bengio. Gradient-based optimization of hyper-parameters. *Neural Computation*, 12(8):1889–1900, 2000.

E. Black, F. Jelinek, J. D. Lafferty, D. M. Magerman, R. L. Mercer, and S. Roukos. Towards history-based grammars: using richer models for probabilistic parsing. In *Meeting of the Association for Computational Linguistics*, pages 31–37, 1993.

R. Bolle, J. Connell, S. Pankanti, N. Ratha, and A. Senior. *Guide to biometrics.* SpringerVerlag, 2003.

Z.I. Botev. A novel nonparametric density estimator. Technical Report, The University of Queensland, 2006.

Thorsten Brants. TnT–A statistical part-of-speech tagger. In *Proceedings of the 6th Applied NLP Conference (ANLP-2000)*, pages 224–231, 2000.

E. Breck, Y.Choi, and C. Cardie. Identifying expressions of opinion in context. In *Proceedings of the Twentieth International Joint Conference on Artificial Intelligence (IJCAI-2007)*, pages 2683–2688, 2007.

L. Breiman. Bias, variance, and arcing classifiers. *Technical Report 460, Statistics Department, University of California*, 1996.

H. Le Cadre. Robust routing and cross-entropy estimation. Manuscript, Technopole Brest Iroise, 2005.

J. Carletta, S. Ashby, S. Bourban, M. Flynn, M. Guillemot, T. Hain, J. Kadlec, V. Karaiskos, W. Kraaij, M. Kronenthal, G. Lathoud, M. Lincoln, A. Lisowska, I. McCowan, W. Post, D. Reidsma, and P. Wellner. The ami meeting corpus: a pre-announcement. In S. Renals and S. Bengio, editors, *MLMI*, volume 3869 of *Lecture Notes in Computer Science*, pages 28–39. Springer, 2005.

W. B. Cavnar and J. M. Trenkle. N-gram-based text categorization. In *Proceedings of SDAIR-94, 3rd Annual Symposium on Document Analysis and Information Retrieval*, pages 161–175, 1994.

C.-C. Chang and C.-J. Lin. *LIBSVM: a library for support vector machines*, 2001. Software available at `http://www.csie.ntu.edu.tw/~cjlin/libsvm`.

O. Chapelle, V. Vapnik, O. Bousquet, and S. Mukherjee. Choosing multiple parameters for Support Vector Machines. *Machine Learning*, 46(1-3):131–159, 2000.

K. Chepuri and T. Homem de Mello. Solving the vehicle routing problem with stochastic demands using the cross entropy method. *Annals of Operations Research*, 134:153–181, 2005.

M. Collins. Head-driven statistical models for natural language parsing. *Computational Linguistics*, 29(4):589–637, 2003.

G.V. Cormack. Harnessing unlabeled examples through iterative application of dynamic Markov modeling. In *Proceedings of the ECML/PKDD 2006 Discovery Challenge*, pages 10–15, 2006.

D. Pereira Coutinho and M. A. T. Figueiredo. Information theoretic text classification using the Ziv-Merhav method. In *IbPRIA (2)*, pages 355–362, 2005.

T. M. Cover and P. E. Hart. Nearest neighbor pattern classification. *IEEE Transactions on Information Theory*, 13:21–27, 1967.

N. Cristianini and J. Shawe-Taylor. *An introduction to Support Vector Machines and other kernel-based learning methods*. Cambridge University Press, New York, NY, USA, 2000.

W. Daelemans and A. van den Bosch. *Memory-based language processing*. Studies in Natural Language Processing. Cambridge University Press, 2005.

W. Daelemans, A. van den Bosch, and J. Zavrel. Forgetting exceptions is harmful in language learning. *Journal of Machine Learning*, 34(1-3):11–41, 1999.

W. Daelemans, V. Hoste, F. De Meulder, and B. Naudts. Combined optimization of feature selection and algorithm parameter interaction in machine learning of language. In *Proceedings of the 14th European Conference on Machine Learning (ECML-2003)*, Lecture Notes in Computer Science 2837, pages 84–95. Springer-Verlag, 2003.

F. Dambreville. Cross-entropic learning of a machine for the decision in a partially observable universe. *Journal of Global Optimization*, 37(4):541–555, 2007.

S. Dasgupta. Learning mixtures of Gaussians. In *Proceedings of the 40th Annual Symposium on Foundations of Computer Science (FOCS'99)*, pages 634–644, 1999.

P. T. de Boer. Analysis and efficient simulation of queueing models of telecommunication systems, PhD thesis, University of Twente, 2000.

P-T. de Boer, D.P. Kroese, S. Mannor, and R.Y. Rubinstein. A tutorial on the cross-entropy method. *Annals of Operations Research*, 134:19–67, 2005.

K. de Jong. *An analysis of the behavior of a class of genetic adaptive systems.* PhD thesis, University of Michigan, 1975.

D. Decoste and D. Mazzoni. Fast query-optimized kernel machine classification via incremental approximate nearest support vectors. In *Proceedings of the 20th International Conference on Machine Learning (ICML'03)*, pages 115–122, 2003.

A. P. Dempster, N. M. Laird, and D. B. Rubin. Maximum likelihood from incomplete data via the EM algorithm. *J. Royal Stat. Soc.*, 39:1–38, 1977.

L. J. P. Van der Maaten. An introduction to dimensionality reduction using Matlab. Technical Report MICC 07-07. Maastricht University, Maastricht, The Netherlands, 2007.

L. J. P. Van der Maaten. *Feature Extraction from Visual Data.* PhD thesis, Tilburg University, 2009.

C. Drummond and R. C. Holte. Cost curves: an improved method for visualizing classifier performance. *Mach. Learn.*, 65(1):95–130, 2006.

R. O. Duda, P. E. Hart, and D. G. Stork. *Pattern classification, Second Edition.* Wiley, 2001.

J. Eisner. Three new probabilistic models for dependency parsing: an exploration. In *Proceedings of the 16th International Conference on Computational Linguistics (COLING-96)*, pages 340–345, 1996.

A. A. Ezhov and G. P. Berman. *Introduction to Quantum Neural Technologies.* Rinton, Princeton (NJ), 2003.

T. Fawcett. An introduction to ROC analysis. *Pattern Recogn. Lett.*, 27(8): 861–874, 2006.

J.G. Fiscus and G. R. Doddington. Topic detection and tracking evaluation overview. In *Topic detection and tracking: event-based information organization*, pages 17–31. Kluwer Academic Publishers, Norwell, MA, USA, 2002.

G. Forman. An extensive empirical study of feature selection metrics for text classification. *Journal of Machine Learning Research*, 3:1289–1305, 2003.

F. Friedrichs and C. Igel. Evolutionary tuning of multiple SVM parameters. *Neurocomputing*, 64:107–117, 2005.

W. A. Gale and G. Sampson. Good-Turing frequency estimation without tears. *Journal of Quantitative Linguistics*, 2(3):217–237, 1995.

M. Galley, K. McKeown, E. Fosler-Lussier, and H. Jing. Discourse segmentation of multi-party conversation. In *Proceedings of the 41st ACL Annual Meeting*, pages 562–569, 2003.

M. Georgescul, A. Clark, and S. Armstrong. An analysis of quantitative aspects in the evaluation of thematic segmentation algorithms. In *Proceedings of the 7th SIGdial Workshop on Discourse and Dialogue*, pages 144–152, 2006.

S. Gerber, T. Tasdizen, and R. T. Whitaker. Robust non-linear dimensionality reduction using successive 1-dimensional Laplacian Eigenmaps. In *Proceedings of the 24th International Conference on Machine Learning (ICML'07)*, pages 281–288, 2007.

C. Giuliano, A. Lavelli, and L. Romano. Exploiting shallow linguistic information for relation extraction from biomedical literature. In *Proceedings of the 9th Conference of the European Chapter of the Association for Computational Linguistics (EACL2006)*, pages 401–408, 2006.

L. K. Grover. A fast quantum mechanical algorithm for database search. In *Proceedings of the 28th Annual ACM Symposium on the Theory of Computing (STOC'96)*, pages 212–219, 1996.

S. Guérif and Y. Bennani. Dimensionality reduction through unsupervised features selection. In *Proceedings of the 10th International Conference on Engineering Applications of Neural Networks (EANN 2007)*, pages 98–106, 2007.

K. Hall and T. Hofmann. Learning curved multinomial subfamilies for natural language processing and information retrieval. In *Proceedings of the Seventeenth International Conference on Machine Learning (ICML'00)*, pages 351–358, 2000.

Z. Harris. *Computable syntactic analysis*, volume 15 of *Transformations and Discourse Analysis Papers*. Philadelphia: University of Pennsylvania, 1959.

M. Hein and M. Maier. Manifold denoising as preprocessing for finding natural representations of data. In *Proceedings of the 21st AAAI Conference on Artificial Intelligence (AAAI-07)*, pages 1646–1649, 2007.

M. Hein, J.-Y. Audibert, and U. Von Luxburg. From graphs to manifolds - weak and strong pointwise consistency of graph Laplacians. In *Proceedings of the 18th Conference on Learning Theory (COLT'05)*, pages 470–485. Springer, 2005.

I. Hendrickx. *Local classification and global estimation, explorations of the k-nearest neighbor algorithm*. PhD thesis, Tilburg University, The Netherlands, 2005.

S. Hettich and S.D. Bay. The UCI KDD Archive, http://kdd.ics.uci.edu, Irvine, CA: University of California, Department of Information and Computer Science, 1999.

Y. Bar Hillel. *Language and information*. Reading, MA: Addison Wesley., 1964.

R. C. Holte and C. Drummond. Cost-sensitive classifier evaluation using cost curves. In T. Washio, E. Suzuki, K. Ming Ting, and A. Inokuchi, editors, *PAKDD*, volume 5012 of *Lecture Notes in Computer Science*, pages 26–29, 2008.

M. Hu and B. Liu. Mining opinion features in customer reviews. In *Proceedings of the 18th AAAI Conference on Artificial Intelligence (AAAI-04)*, pages 755–760, 2004.

D. A. Hull and S. E. Robertson. The TREC-8 filtering track final report. In *Proceedings of TREC*, 1999.

M. Jammer. *The conceptual development of quantum mechanics*. McGraw-Hill, New York, 1966.

T. Jebara. *Discriminative, generative and imitative learning*. PhD thesis, MIT, 2001.

T. Jebara. *Machine learning : discriminative and generative*. Springer, 2003.

T. Jebara, R. Kondor, and A. Howard. Probability product kernels. *Journal of Machine Learning Research*, 5:819–844, 2004.

F. Jelinek. *Statistical models for speech recognition*. MIT Press, 1997.

T. Joachims. Text categorization with Support Vector Machines: learning with many relevant features. In *Proceedings of the 10th European Conference on Machine Learning (ECML-98)*, pages 137–142, 1998.

T. Joachims. *Learning to classify text using Support Vector Machines*. Kluwer, 2002.

T. Joachims. Transductive inference for text classification using Support Vector Machines. In Ivan Bratko and Saso Dzeroski, editors, *Proceedings of the 16th International Conference on Machine Learning (ICML'99)*, pages 200–209. Morgan Kaufmann Publishers, San Francisco, US, 1999.

T. Joachims. SVMlight, http://svmlight.joachims.org/, 2004.

K. Sparck Jones and J. R. Galliers. Evaluating natural language processing systems: an analysis and review. *Machine Translation*, 12(4):375–379, 1997.

I. Kanaris and E. Stamatatos. Webpage genre identification using variable-length character n-grams. In *Proceedings of the 19th International Conference on Tools with Artificial Intelligence (ICTAI 2007)*, pages 3–10, 2007.

S. Kasif, S. Salzberg, D. L. Waltz, J. Rachlin, and D. W. Aha. A probabilistic framework for memory-based reasoning. *Artificial Intelligence*, 104(1-2):287–311, 1998.

R. E. Kass. The geometry of asymptotic inference. *Statistical Science*, 4(3): 188–234, 1989.

S. M. Katz. Estimation of probabilities from sparse data for the language model component of a speech recogniser. *IEEE Transactions on Acoustics, Speech, and Signal Processing*, 35(3):400–401, 1987.

A. Kennedy and D. Inkpen. Sentiment classification of movie reviews using contextual valence shifters. *Computational Intelligence*, 22(2):110–125, May 2006.

S.-M. Kim and E. Hovy. Determining the sentiment of opinions. In *Proceedings of the 42th Annual Meeting on Association for Computational Linguistics (ACL'04)*, pages 1367–1373, 2004.

S.-M. Kim and E. Hovy. Automatic identification of pro and con reasons in online reviews. In *Proceedings of the 44th Annual Meeting on Association for Computational Linguistics (ACL'06)*, pages 483–490, 2006.

R. Kohavi. A study of cross-validation and bootstrap for accuracy estimation and model selection. In *Proceedings of the Fourteenth International Joint Conference on Artificial Intelligence (IJCAI)*, pages 1137–1145, 1995.

R. Kohavi and G. John. Wrappers for feature subset selection. *Artificial Intelligence journal, special issue on relevance*, 97(1-2):273–324, 1997.

R. Kohavi and D. H. Wolpert. Bias plus variance decomposition for zero-one loss functions. In Lorenza Saitta, editor, *Proceedings the 13th International Conference on Machine Learning (ICML'96)*, pages 275–283. Morgan Kaufmann, 1996.

T. Kohonen. *Self-Organizing Maps*. Springer Series in Information Sciences, Vol. 30, Springer, Berlin, Heidelberg, New York, 2001.

W. Kraaij. *Variations on language modeling for information retrieval*. PhD thesis, University of Twente, 2004.

W. Kraaij, S. Raaijmakers, and P. Elzinga. Maximizing classifier utility for a given accuracy. In *Proceedings of the 20th Belgian-Dutch Conference on Artificial Intelligence (BNAIC-2008)*, pages 121–128, 2008.

G. Kuperberg. A subexponential-time quantum algorithm for the dihedral hidden subgroup problem. *SIAM J. Comput.*, 35:170–188, 2005.

J. Lafferty and G. Lebanon. Diffusion kernels on statistical manifolds. *Journal of Machine Learning*, 6:129–163, 2005.

G. Lebanon. Learning Riemannian metrics. In *Proceedings of the 19th Conference on Uncertainty in Artificial Intelligence, AUAI Press*, pages 362–369, 2003.

G. Lebanon. Information geometry, the embedding principle, and document classification. In *Proceedings of the 2nd International Symposium on Information Geometry and its Applications*, pages 101–108, 2005.

G. Lebanon. Metric learning for text documents. *IEEE Trans. Pattern Anal. Mach. Intell.*, 28(4):497–508, 2006a.

G. Lebanon. Sequential document representations and simplicial curves. In *Proceedings of the 22nd Conference on Uncertainty in Artificial Intelligence*, pages 273–280, 2006b.

J. M. Lee. *Riemannian manifolds: an introduction to curvature*. Springer, 1997.

C. Leslie and R. Kuang. Fast string kernels using inexact matching for protein sequences. *J. Mach. Learn. Res.*, 5:1435–1455, 2004.

D. D. Lewis. Evaluating and optimizing autonomous text classification systems. In *Proceedings of the 18th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval (SIGIR '95)*, pages 246–254. ACM Press, 1995.

X. Li and D. Roth. Exploring evidence for shallow parsing. In *Proceedings of the Annual Conference on Computational Natural Language Learning*, pages 1–7, 2001.

Y. Li, K. Bontcheva, and H. Cunningham. Experiments of Opinion analysis on the Corpora MPQA AND NTCIR-6. In *Proceedings of the NTCIR-6 Workshop Meeting*, pages 323–329, 2007.

J. Lin and D. Demner-Fushman. Evaluating summaries and answers: two sides of the same coin? In *Proceedings of the ACL Workshop on Intrinsic and Extrinsic Evaluation Measures for Machine Translation and/or Summarization*, pages 41–48. Association for Computational Linguistics, 2005.

Y. Lu, I. Cohen, X. Sean Zhou, and Q. Tian. Feature selection using principal feature analysis. In *Proceedings of the 15th international conference on Multimedia (MULTIMEDIA'07)*, pages 301–304, 2007.

H. P. Luhn. A statistical approach to mechanized encoding and searching of literary information. *IBM Journal of Research and Development*, 4(2):309–317, May 1957.

S. A. Macskassy, H. Hirsh, A. Banerjee, and A. A. Dayanik. Using text classifiers for numerical classification. In *Proceedings of the 17th International Joint Conference on Artificial Intelligence (IJCAI-01)*, pages 885–890, 2001.

A. Martin, G. Doddington, T. Kamm, M. Ordowski, and M. Przybocki. The DET curve in assessment of detection task performance. In *Proceedings of Eurospeech '97*, pages 1895–1898, 1997.

A. F. T. Martins, M. A. T. Figueiredo, P. M. Q. Aguiar, N. A. Smith, and E. P. Xing. Nonextensive entropic kernels. In *Proceedings of the 25th international conference on Machine learning (ICML'08)*, pages 640–647, 2008.

P. Mcnamee, J. Mayfield, and C. Piatko. The haircut system at TREC-9. In *Proceedings of the Ninth Text REtrieval Conference (TREC-9), NIST Special Publication 500-249*, pages 273–279, 2001.

N. David Mermin. *Quantum computer science: an introduction*. Cambridge University Press, New York, NY, USA, 2007.

P. Mittelstaedt, A. Prieur, and R. Schieder. Unsharp particle-wave duality in a photon split-beam experiment. *Foundations of Physics*, 17:891–903, 1987.

P. J. Moreno, P.P. Ho, and N. Vasconcelos. A Kullback-Leibler divergence based kernel for SVM classification in multimedia applications. In *In Advances in Neural Information Processing Systems 16*, pages 1385–1392. MIT Press, 2003.

T. Mullen and N. Collier. Sentiment analysis using Support Vector Machines with diverse information sources. In *EMNLP*, pages 412–418, 2004.

V. Ng, S. Dasgupta, and S. M. Niaz Arifin. Examining the role of linguistic knowledge sources in the automatic identification and classification of reviews. In *Proceedings of the 21st International Conference on Computational Linguistics (COLING-06)*, pages 611–618, 2006.

B. Pang and L. Lee. A sentimental education: sentiment analysis using subjectivity summarization based on minimum cuts. In *Proceedings of the ACL*, pages 271–278, 2004.

B. Pang, L. Lee, and S. Vaithyanathan. Thumbs up? sentiment classification using machine learning techniques. In *Proceedings of the 2002 Conference on Empirical Methods in Natural Language Processing (EMNLP-2002)*, pages 79–86, 2002.

L. Pevzner and M. Hearst. A critique and improvement of an evaluation metric for text segmentation. *Computational Linguistics*, 28 (1):19–36, 2002.

J. Platt. Probabilistic outputs for support vector machines and comparison to regularized likelihood methods. In A.J. Smola, P. Bartlett, B. Schölkopf, and D. Schuurmans, editors, *Advances in Large Margin Classiers*, pages 61–74, 2000.

J. Poelmans, P. Elzinga, S. Viaene, and G. Dedene. An exploration into the power of formal concept analysis for domestic violence analysis. In *Proceedings of the 8th Industrial Conference on Data Mining (ICDM 2008)*, volume 5077, pages 404–416, 2008.

J. Ross Quinlan. *C4.5: Programs for machine learning*. Morgan Kaufmann, 1993.

R. Quirk, S. Greenbaum, G. Leech, and J. Svartvik. *A comprehensive grammar of the English language*. Longman, New York, 1985.

S. Raaijmakers. Sentiment classification with interpolated information diffusion kernels. In *Proceedings of the First International Workshop on Data Mining and audience Intelligence for advertising (ADKDD'07)*, 2007.

S. Raaijmakers. Finding representations for memory-based language learning. In *Proceedings of The Third Workshop on Computational Language Learning (CoNLL-1999)*, pages 24–32, 1999.

S. Raaijmakers and W. Kraaij. A shallow approach to subjectivity classification. In *Proceedings of ICWSM*, 2008.

S. Raaijmakers and W. Kraaij. Polarity classification of blog TREC 2008 data with geodesic kernels. In *Proceedings TREC 2008, Gaithersburg, USA*, 2009.

S. Raaijmakers and T. Wilson. Comparing word, character, and phoneme n-grams for subjective utterance recognition. In *Proceedings Interspeech*, 2008.

S. Raaijmakers, K. Truong, and T. Wilson (2008). Multimodal subjectivity analysis of multiparty conversation. In *Proceedings EMNLP'08*, pages 466–474, 2008.

L. A. Ramshaw and M. P. Marcus. Text chunking using transformation-based learning. In *Proceedings of the Third Annual Workshop on Very Large Corpora (WVLC-3)*, pages 82–94. ACL, 1995.

A. Ratnaparkhi. *Maximum entropy models for natural language ambiguity resolution*. PhD thesis, Computer and Information Science, University of Pennsylvania, 1998a.

A. Ratnaparkhi. Statistical models for unsupervised prepositional phrase attachment. In *Proceedings of the 17th International Conference on Computational Linguistics (COLING-98)*, pages 1079–1085, 1998b.

B. Ricks and D. Ventura. Training a quantum neural network. In *Neural Information Processing Systems*, pages 1019–1026, December 2003.

E. Riloff, J. Wiebe, and T. Wilson. Learning subjective nouns using extraction pattern bootstrapping. In *Proceedings of The Seventh Workshop on Computational Language Learning (CoNLL-2003)*, pages 25–32, 2003.

E. Riloff, J. Wiebe, and W. Phillips. Exploiting subjectivity classification to improve information extraction. In *Proceedings of the 19th AAAI Conference on Artificial Intelligence (AAAI-05)*, pages 1106–1111, 2005.

E. Riloff, S. Patwardhan, and J. Wiebe. Feature subsumption for opinion analysis. In *Proceedings of the 2006 Conference on Empirical Methods in Natural Language Processing (EMNLP-2006)*, pages 440–448, July 2006.

J. Rissanen. A universal prior for integers and estimation by minimum description length. *Annals of Statistics*, 11(2):416–431, 1983.

S. E. Robertson and K. Sparck Jones. Relevance weighting of search terms. *Journal of the American Society for Information Science*, 27(3):129–146, 1976.

F. Rosenblatt. The perceptron: a probabilistic model for information storage and organization in the brain. *Psychological Review*, 65(6):386–408, 1958.

R. Rosenfeld. Two decades of statistical language modeling: where do we go from here? In *Proceedings of the IEEE, 88(8)*, pages 1270–1278, 2000.

R. Y. Rubinstein. The cross-entropy method for combinatorial and continuous optimization. *Methodology and Computing in Applied Probability*, 2:127–190, 1999.

G. Salton, A. Wong, and C. S. Yang. A vector space model for automatic indexing. *Communications of the ACM*, 18(11):613–620, 1975.

B. Schölkopf and A. J. Smola. *Learning with kernels*. MIT Press, 2002.

Y. Seki, K. Eguchi, N. Kando, and M. Aono. Multi-document summarization with subjectivity analysis at DUC 2005. In *Proceedings of the 2005 Document Understanding Conference (DUC-2005)*, 2005.

B. Settles. ABNER: an open source tool for automatically tagging genes, proteins, and other entity names in text. *Bioinformatics*, 21(14):3191–3192, 2005.

R. N. Shepard. Toward a universal law of generalization for psychological science. *Science*, 237:1317–1323, 1987.

P. Shor. Polynomial-time algorithms for prime factorization and discrete logarithms on a quantum computer. *SIAM J. Comput.*, 26:1484–1509, 1997.

W. Siedlecki and J. Sklansky. A note on genetic algorithms for large-scale feature selection. *Pattern Recogn. Lett.*, 10(5):335–347, 1989.

R. W. Sinnott. Virtues of the haversine. *Sky and Telescope*, 68(2):159, 1984.

E. Stamatatos. Ensemble-based author identification using character n-grams. In *Proceedings of the 3rd Int. Workshop on Text-based Information Retrieval (TIR'06)*, pages 41–46, 2006.

C. Stanfill and D. Waltz. Toward memory-based reasoning. *Communications of the ACM*, 29(12):1213–1228, 1987.

V. Stoyanov and C. Cardie. Toward opinion summarization: linking the sources. In *Proceedings of the Workshop on Sentiment and Subjectivity in Text*, pages 9–14, 2006.

D. M. J. Tax. *One-class classification; concept-learning in the absence of counter-examples*. PhD thesis, Delft University of Technology, 2001.

E. F. Tjong Kim Sang and F. De Meulder. Introduction to the CoNLL-2003 shared task: Language-Independent Named Entity Recognition. In Walter Daelemans and Miles Osborne, editors, *Proceedings of The Seventh Workshop on Computational Language Learning (CoNLL-2003)*, pages 142–147, 2003.

K. Truong and S. Raaijmakers. Recognition of emotion in spontaneous emotional speech using acoustic and lexical features. In *Proceedings of the 5th Joint Workshop on Machine Learning and Multimodal Interaction, Utrecht, The Netherlands*, pages 161–172, 2008.

P. Turney. Thumbs up or thumbs down? Semantic orientation applied to unsupervised classification of reviews. In *Proceedings of the Association for Computational Linguistics (ACL)*, pages 417–424, 2002.

D. G. Underhill, L. McDowell, D. J. Marchette, and J. L. Solka. Enhancing text analysis via dimensionality reduction. In *Proceedings of the IEEE International Conference on Information Reuse and Integration (IRI 2007)*, pages 348–353, 2007.

A. van den Bosch. Wrapped progressive sampling search for optimizing learning algorithm parameters. In R. Verbrugge, N. Taatgen, and L. Schomaker, editors, *Proceedings of the 16th Belgian-Dutch Conference on Artificial Intelligence (BNAIC-2004)*, pages 219–226, 2004.

A. van den Bosch and S. Buchholz. Shallow parsing on the basis of words only: a case study. In *Proceedings of the 40th Annual Meeting on Association for Computational Linguistics (ACL'02)*, pages 433–440, 2001.

K. van Rijsbergen. *The Geometry of Information Retrieval*. Cambridge University Press, 2004.

K. van Rijsbergen. *Information retrieval*. Butterworths, 1979.

V. N. Vapnik. *The nature of statistical learning theory*. Springer, 2 edition, 1999.

D. Ventura and T. Martinez. Quantum associative memory. *Information Sciences*, 124(1-4):273–296, 2000.

S. Verberne, D. Theijssen, L. Boves, and S. Raaijmakers. Learning to rank answers to why-questions. In *Proceedings of the 9th Dutch-Belgian Information Retrieval Workshop (DIR'09)*, pages 34–42, 2009.

J. Watrous. Quantum computational complexity. In *Encyclopedia of Complexity and System Science*, 2009.

G. I. Webb. MultiBoosting: a technique for combining boosting and wagging. *Machine Learning*, 40(2):159–196, 2000.

J. Wiebe and R. Mihalcea. Word sense and subjectivity. In *Proceedings of the 21st International Conference on Computational Linguistics and the 44th annual meeting of the ACL (ACL'06)*, pages 1065–1072, 2006.

J. Wiebe, T. Wilson, and C. Cardie. Annotating expressions of opinions and emotions in language. *Language resources and evaluation*, 39(2-3):165–210, 2005.

J. M. Wiebe, R. F. Bruce, and T. P. O'Hara. Development and use of a gold-standard data set for subjectivity classifications. In *Proceedings of the 37th annual meeting of the Association for Computational Linguistics on Computational Linguistics*, pages 246–253, 1999.

T. Wilson. Annotating subjective content in meetings. In *Proceedings of the Sixth International Language Resources and Evaluation (LREC'08)*, 2008.

T. Wilson, J. Wiebe, and P. Hoffmann. Recognizing contextual polarity in phrase-level sentiment analysis. In *Proceedings of the conference on Human Language Technology and Empirical Methods in Natural Language Processing (HLT/EMNLP 2005)*, pages 347–354, 2005.

T. Wilson, J. Wiebe, and R. Hwa. Recognizing strong and weak opinion clauses. *Computational Intelligence*, 22(2):73–99, 2006.

J. Yang, Y. Jiang, A. G. Hauptmann, and C. Ngo. Evaluating bag-of-visual-words representations in scene classification. In *Proceedings of the international workshop on Workshop on multimedia information retrieval (MIR'07)*, pages 197–206, 2007.

Y. Yang and J. O. Pedersen. A comparative study on feature selection in text categorization. In *Proceedings of the 14th International Conference on Machine Learning (ICML'97)*, pages 412–420, 1997.

H. Yu and V. Hatzivassiloglou. Towards answering opinion questions: separating facts from opinions and identifying the polarity of opinion sentences. In *Proceedings of the 2003 Conference on Empirical Methods in Natural Language Processing (EMNLP-2003)*, pages 129–136. Association for Computational Linguistics, 2003.

D. Zhang and W. Sun Lee. Extracting key-substring-group features for text classification. In *Proceedings of the Twelfth ACM SIGKDD International Conference on Knowledge Discovery and Data Mining (KDD'06)*, pages 474–483, 2006.

D. Zhang, X. Chen, and W. Sun Lee. Text classification with kernels on the multinomial manifold. In *Proceedings SIGIR'05*, pages 266–273, 2005.

H. Zhang, A. C. Berg, M. Maire, and J. Malik. Svm-knn: Discriminative nearest neighbor classification for visual category recognition. In *Proceedings of the 2006 IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR'06)*, pages 2126–2136, 2006.

J. Zhao, K. Lu, and X. He. Locality sensitive semi-supervised feature selection. *Neurocomput.*, 71(10-12):1842–1849, 2008.

Appendix A

Information geometry

A.1 The simplex

An n-simplex is the n-dimensional version of a triangle. It is usually represented in $n+1$-dimensional space, and it is defined as

$$\Delta^n = \{(x_1, \ldots, x_n) \in \mathbb{R}^{n+1} \mid \sum_i x_i = 1\} \tag{A.1}$$

For instance, depicted in figure A.1, is the 2-simplex, the set $\{(p_1, p_2, p_3)\} \subseteq \mathbb{R}^3$. L1-normalized data naturally forms a simplex of degree n for n-dimensional

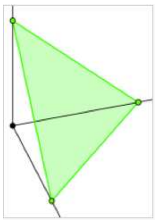


Figure A.1: The 2d-simplex represented in 3d-space.

data.

A.2 Manifolds

Differentiable or smooth manifolds are topological spaces in which every point has a local Euclidean neighborhood, but the relation between points in general is not necessarily Euclidean. A trivial example is our globe. Two points on the globe that are very close to each other can be assumed to have a Euclidean

relationship, but for points that are far apart, the curvature of the globe should be taken into account for accurate distance measurement. Put differently, for estimating the distance between local points, Euclidean distance, a straight line segment, is a good approximation. Distance between points much further apart should be measured using geodesics: the shortest curve that connects them. Formally, a topological space is a set of points P paired with a function $F : POW(P) \mapsto \{0, 1\}$, which is closed under union and intersection.

- $F(\emptyset) = 1; F(S) = 1$
- $\forall\ s_i \in F : F(\cup_i s_i) = 1$
- $\forall\ s_i \in F : F(\cap_i s_i) = 1$

If every point in the topological space has a local neighborhood then the space is a manifold. A local neighborhood here means a collection of points on an n-dimensional disk with a certain radius; if $n > 3$, the disk is actually a ball. For this collection to qualify as a neighborhood for a point x, x must be the center of the disk. A differentiable manifold that is equipped with a metric tensor or Riemannian metric is called a *Riemannian manifold*. A Riemannian metric g is a symmetric and positive definite distance measure:

$$g(X, Y) = g(Y, X)g(X, X) > 0 \text{ if } X \neq 0 \tag{A.2}$$

A *diffeomorphism* is a map between two manifolds that is differentiable (i.e. for every point in the domain, the derivative exists), and has a differentiable inverse. This is basically the equivalent of a bijective, continuous, differentiable function. Given $F : M \mapsto N$, a diffeomorphism between two manifolds M and N. The tangent space T_xM is the set of curves through the point $x \in M$, under a suitable equivalence relation. These curves are called *tangent vectors*, and can be interpreted as velocities. With F, one can associate a *push-forward* map F_* mapping the tangent spaces T_xM, T_yM to T_xN, T_yN:

$$F_* : T_xM \mapsto T_{F(x)}N \tag{A.3}$$

This is a map from velocity curves in one space to velocity curves in a space associated through a diffeomorphism with the original space. A Riemannian metric g on a manifold assigns to every point $x \in M$ the inner product of the tangent space T_xM. The length of a tangent vector $v \in T_xM$ can now be defined as $\parallel v \parallel_x = \sqrt{g_x(v, v)}$. The length of a curve c between points a and b is defined as

$$L(c) : [a, b] \mapsto M = \int_a^b \parallel c'(t) \parallel dt \tag{A.4}$$

with c' the first derivative of c. The geodesic distance between two points a and b on M is the length of the shortest curve connecting a and b. The combination

of M and a geodesic distance measure turns M into a *metric space*. Given such a distance measure, h, the *pullback* F^*h is defined as

$$T^*_{F(p)}N \mapsto T^*_pM \tag{A.5}$$

This triggers a *pullback metric*

$$F^*h_x(u,v) = h_{F(x)}(F_*(u), F_*(v)) \tag{A.6}$$

Here $h_x(u,v)$ is the function that returns the inner product of u and v if these are in the tangent space of x (and zero else). We obtain *isometry* as follows:

$$d_{F^*h}(x,y) = d_h(F(x), F(y)) \tag{A.7}$$

Isometry means that the distances in the pushed forward and pulled-back spaces are related through a diffeomorphism.

Appendix B

Classifier evaluation

Classifiers can be evaluated from several perspectives. Historically, the standard perspective is *intrinsic*: from the perspective of a number of datasets, and a specific classifier setup. As formulated by Sparck Jones and Galliers (Jones and Galliers [1997]), intrinsic criteria are related to a system's objective, and extrinsic criteria apply to its function: its role in relation to its intended use. Extrinsic measures often evaluate classifier performance with task-specific penalty systems.

A generic evaluation procedure for classifiers is *cross-validation* (e.g. Kohavi [1995]), where a data set is split into a number of training-test partitionings called *folds*. Subsequently, a classifier is trained on every training partition and its corresponding test partition, after which performance is measured. The separate performances are averaged upon completion. For instance, a 10-fold cross-validation would partition a data set into 10 training-test folds, and run 10 experiments.

Familiar intrinsic evaluation measures are accuracy, recall and precision, and combinations of recall and precision such as F-score. Some extrinsic measures are linear utility and detection cost.

Intrinsic measures

Given a binary classifier C that is applied to a labeled test set T, we define in table B.1 a number of basic notions. From these basic ingredients, the familiar notions of accuracy, recall, and precision can be computed:

$$\begin{aligned} &Accuracy = \frac{TP}{TP+FP+TN+FN} \\ &Recall\ (True\ Positive\ Rate,\ Sensitivity) = \frac{TP}{TP+FN} \\ &Precision\ or\ Positive\ Predictive\ Value\ (PPV) = \frac{TP}{TP+FP} \end{aligned} \tag{B.1}$$

Ground truth	Prediction	Definition	
+	+	true positive	TP
-	+	false positive	FP
-	-	true negative	TN
+	-	false negative	FN

Table B.1: TP, FP, TN, FN under a binary classifier setting, from the perspective of the positive class.

The F-score, a harmonic mean of Recall and Precision, is defined as (van Rijsbergen [1979], Lewis [1995])

$$F_\beta = \frac{(\beta^2 + 1) * Precision * Recall}{\beta^2 * Precision + Recall} \tag{B.2}$$

or equivalently

$$F_\beta = \frac{(\beta^2 + 1) * TPR * PPV}{\beta^2 * TPR + PPV} \tag{B.3}$$

where β usually is 1[1]. Notice that the F-score only attains a high value if both Recall and Precision are high. If $\beta = 1$, then Recall and Precision are treated as equally important. If $\beta = 0.5$, then Precision is twice as important as Recall.

This view on the relationship between ground truth and predictions is purely from the positive class. The corresponding version of F_β is the so-called *micro-averaged* version, which typically is computed only for one class (the positive) for a binary classifier (i.e. $\mid C \mid$, the number of classes, is treated as 1):

$$\begin{aligned} F_{micro} &= \tfrac{(\beta^2+1)Precision_{micro} * Recall_{micro}}{\beta^2 * Precision_{micro} + Recall_{micro}} \\ Precision_{micro} &= \tfrac{\sum_{i=1}^{|C|} TP_i}{\sum_{i=1}^{|C|} TP_i + FP_i} \\ Recall_{micro} &= \tfrac{\sum_{i=1}^{|C|} TP_i}{\sum_{i=1}^{|C|} TP_i + FN_i} \end{aligned} \tag{B.4}$$

If we take the dual perspective from the negative class, then a prediction '-' for a '-' ground truth class is equally a 'true positive': a correct prediction for the negative class.

Translating the definitions of Recall, Precision and F-score, we end up with Recall and Precision from the perspective of the dual (negative) class:

$$Recall_{neg} = Negative\ Predictive\ Value\ (NPV) = \frac{TN}{TN + FN} \tag{B.5}$$

[1] In TREC 2003, β was 5, and in TREC 2004, β was 3 (see for some historical context e.g. Lin and Demner-Fushman [2005]).

Ground truth	Prediction	Definition	
+	+	true negative	TN
-	+	false negative	FN
-	-	true positive	TP
+	-	false positive	FP

Table B.2: TP, FP, TN, FN under a binary classifier setting, from the perspective of the negative class.

$$Precision_{neg} = Specificity_{neg} = \frac{TN}{TN + FP} \tag{B.6}$$

The quantity

$$False\ Positive\ rate\ (FPR) = \frac{FP}{FP + TN} \tag{B.7}$$

leads to the alternative definition of Specificity as

$$Specificity_{neg} = 1 - FPR \tag{B.8}$$

From this shifted class perspective we obtain an alternative F-measure from the perspective of the negative class:

$$F_{neg} = \frac{(\beta^2 + 1) * NPV * Specificity_{neg}}{\beta^2 * NPV + Specificity_{neg}} \tag{B.9}$$

When we average the two F-measures, we obtain a *macro-averaged*, two-class version of the F-measure:

$$\begin{aligned} F_{macro} = \quad & \tfrac{1}{2}\left[\tfrac{(\beta^2+1)*TPR*PPV}{\beta^2*TPR+PPV} + \tfrac{(\beta^2+1)*NPV*Specificity_{neg}}{\beta^2*NPV+Specificity_{neg}}\right] = \\ & \tfrac{1}{2}(\beta^2+1)\left[\tfrac{TPR*PPV}{\beta^2*TPR+PPV} + \tfrac{NPV*Specificity_{neg}}{\beta^2*NPV+Specificity_{neg}}\right] = \\ & \tfrac{(\beta^2+1)}{2}\left[\tfrac{TPR*PPV}{\beta^2*TPR+PPV} + \tfrac{NPV*Specificity_{neg}}{\beta^2*NPV+Specificity_{neg}}\right] \end{aligned} \tag{B.10}$$

Generalizing this to the multiclass setting, with more than 2 classes, table (B.3) lists class-specific contingencies. We then define class-specific versions of Precision and Recall:

$$\begin{aligned} Precision_c &= \tfrac{TP_c}{TP_c+FP_c} \\ Recall_c &= \tfrac{TP_c}{TP_c+FN_c} \end{aligned} \tag{B.11}$$

Micro-averaging precision and recall is based on global counts for TP, FP, and FN, after which precision and recall are computed, whereas macro-averaging

Ground truth	Prediction	Definition	
c	c	true positive for c	TP_c
not c	c	false positive for c	FP_c
not c	not c	true negative for c	TN_c
c	not c	false negative for c	FN_c

Table B.3: TP, FP, TN, FN under a multiclass classifier setting.

involves first computing precision and recall per class, and then averaging the results. The latter approach is especially suited for class-imbalanced datasets, as all class-specific scores for precision and recall are equally weighted. Micro-averaging is a form of *document-pivoted* evaluation, whereas macro-averaging can be seen as *category-pivoted*.

The macro-averaged F-score then is defined as

$$
\begin{aligned}
F_{macro} &= \tfrac{(\beta^2+1)*Precision_{macro}*Recall_{macro}}{\beta^2*Precision_{macro}+Recall_{macro}} \\
Precision_{macro} &= \tfrac{1}{|C|}\sum_{i=1}^{|C|}\frac{TP_i}{TP_i+FP_i} \\
Recall_{macro} &= \tfrac{1}{|C|}\sum_{i=1}^{|C|}\frac{TP_i}{TP_i+FN_i}
\end{aligned}
\tag{B.12}
$$

Mean averaged precision (MAP) is a measure typically used in a ranked document retrieval task, where the average precision for every query is computed, and the mean is taken over all queries. This effectively is macro-averaging: all queries are considered equal. Precision at R (R-PREC) is the precision when exactly R documents are retrieved.

ROC curves

ROC curves plot the True Positive Rate (TPR) (y-axis) against the False Positive Rate (FPR)(x-axis). A common aggregate statistic for measuring classification is AUC: the *Area Under (ROC) Curve* (Fawcett [2006]). Discrete classifiers emit symbolic classifications, and therefore the quantities TPR and FPR are single numbers, which together produce just one point in ROC space (figure B.1(a)). This point can be seen as an optimal operating point: the optimal classifier attains maximal performance at the point (TPR, FPR). The theoretically best performing classifier has TPR=1, and FRP=0. Therefore, any ROC curve should have its peak as close to the left top corner (the point $(0, 1)$) as possible. For binary classifiers that produce continuous output, such

as class probabilities, or decision values, the output can be thresholded (above a certain value produce class +1, below produce −1) by varying classifier-specific parameters. This will produce a smooth ROC curve (figure B.1(b))

DET curves

Whereas ROC curves only plot one type of error (the False Alarm Rate), Detection Error Tradeoff (DET) curves (Martin et al. [1997]) depict the tradeoff between two errors: P_{miss} and P_{FA}. This makes this type of plot more suitable for assessing the performance of a classifier when a trade-off between two error types needs to be investigated. A sample DET curve comparing two systems is given in figure B.2. Optimal curves are skewed towards the bottom left.

Extrinsic measures

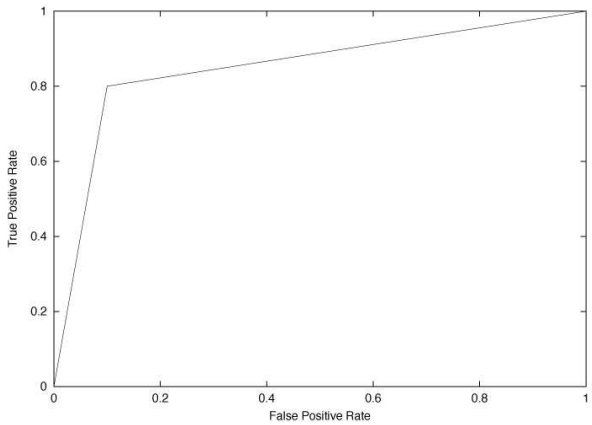
The TREC filtering task used a *linear utility function* for the adaptive filtering task, which is a rather complex classification task where a system can use feedback in order to set its optimal operating point (decision threshold) in a dynamic fashion. The linear utility is defined as Hull and Robertson [1999]:

$$linear\ utility = \alpha\, TP + \beta FP + \gamma FN + \delta\, TN \tag{B.13}$$

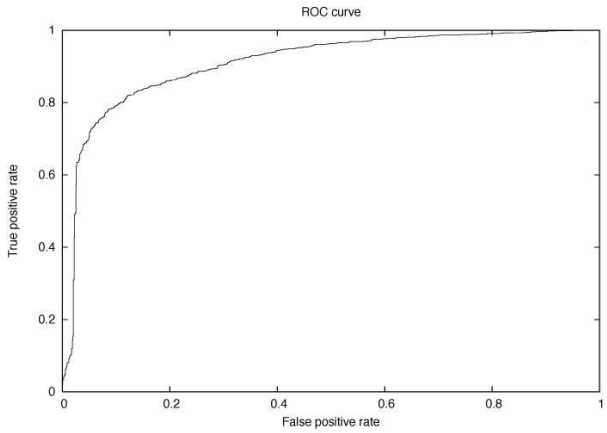
This is essentially a cost function, where parameters must be chosen to fit a particular user scenario. A more elegant way to model the *cost* of running a certain classifier on a dataset is the family of cost functions that were developed in the Topic Detection and Tracking (TDT) framework Fiscus and Doddington [2002]. The basic *detection cost* function is defined as follows:

$$C_{det} = C_{Miss} \times P_{Miss} \times P_{target} + C_{FA} * P_{FA} * (1 - P_{target}) \tag{B.14}$$

where C_{Miss} and C_{FA} are fixed cost parameters that tax type II and type I errors respectively (False Negatives and False Positives), P_{Miss} and P_{FA} are the system-dependent probabilities of type II and type I errors, and $1 - P_{target}$ is the prior probability of a positive class label *T=target*. Usually, the detection cost is measured at different levels of Miss/False Alarm trade-off by threshold sweeping, thus generating a detection cost curve. The detection cost function is motivated by the desire to quantify different types of error and sum the complete cost of a detection task for a certain data collection (taking into account the relative proportion of the class population sizes). However, the detection cost is based on a fully automatic scenario. Incorporating the cost of manually assessing observations would make the detection cost function less intuitive.



(a) Sample ROC curve based on a single point in ROC space.



(b) Sample ROC curve based on SVM decision values.

Figure B.1: ROC plots.

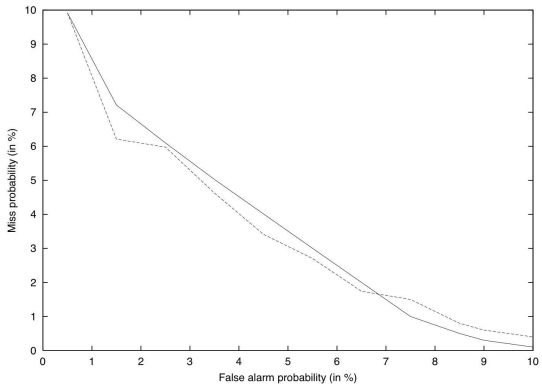


Figure B.2: A sample DET curve comparing two systems.

Cost curves

As pointed out by Drummond and Holte [2006], ROC curves are not adequate for cost-based evaluation of the performance of a classifier. Also, the performance differences between two classifiers are not easily read off from ROC curves, as the ROC curves may cross on several points. The AUC measure partially meets this problem, by integrating over the area under the ROC curve, which should be maximal for best performing classifiers. Cost curves on the other hand (Drummond and Holte [2006]) do allow for a penalty-based evaluation of a classifier's performance, which makes comparison of different classifiers transparent and better motivated from a statistical point of view by allowing for confidence intervals related to performance. Given a binary classification problem, a cost curve plots the *probability cost* for the positive class (+) against the *normalized expected cost*. Let $p(+)$ be the a priori probability of observing an element of the positive class in the training data. The cost terms $C(+ \mid -)$ and $C(- \mid +)$ represent the costs of misclassification (a $-$ as a $+$, and vice versa). These costs are task-dependent. The unbiased view is that they are both 0.5. The Probability Cost for the positive class, $PC(+)$, aggregates these quantities as follows:

$$PC(+) = \frac{p(+)C(- \mid +)}{p(+)C(- \mid +) + (1 - p(+))C(+ \mid -)} \tag{B.15}$$

This is the cost component (a percentage) that is represented by misclassifying positive cases as negative (type II errors). The actual classifier performance

quantity is the Normalized Expected Cost (NEC), defined as

$$NEC = FN * PC(+) + FP * (1 - PC(+)) \tag{B.16}$$

Setting $PC(+) = 0$ for $x = 0$, and $PC(+) = 1$ for $y = 1$, we obtain two points: $(0, FP)$ and $(1, FN)$. A cost curve (Holte and Drummond [2008]) for a classifier is simply drawn by connecting these two points with a straight line, after which the normalized expected cost for the classifier can be read off directly. Figure B figure shows an example. Lower lines indicate better performance.

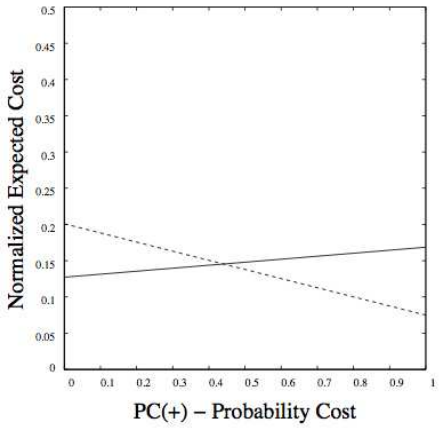


Figure B.3: A sample cost curve.

The line from $(0, 0)$ to $(1, 1)$ corresponds to a trivial classifier always assigning the negative class. The line from $(0, 1)$ to $(1, 0)$ corresponds to the dual trivial classifier that always assigns the positive class. See figure B.4; the optimal performance for the classifier depicted with the lowest line, compared to the two trivial classifiers, happens in the interval $0.18 < PC(+) < 0.85$.

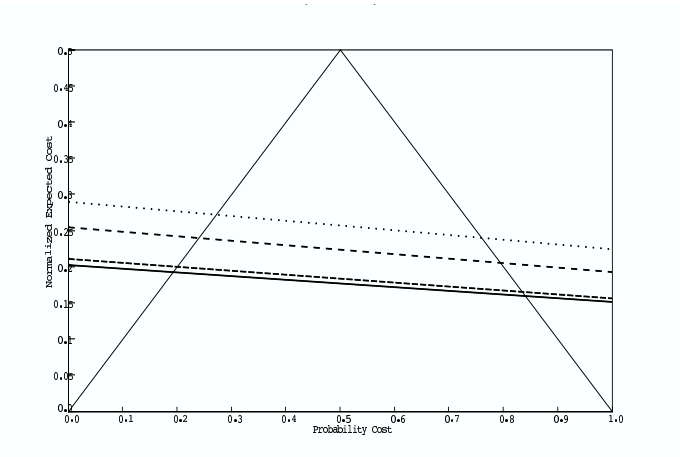


Figure B.4: Combined cost curves for 4 classifiers, and two trivial classifiers.

Appendix C

Algorithms

Algorithm C.1 General CE algorithm for rare event simulation.

Choose initial parameters N, ρ. Choose an initial parameter vector u.
Define $\hat{v}_0 = u$. Set $t = 1$.
Generate a random sample $X_1, \ldots, X_n$ according to the pdf $f(\cdot; \hat{v}_{t-1})$.
Compute the performance $S(X_i)$ for every i and rank results $S_{(1)} \leq \ldots \leq S_{(n)}$.
Compute $\gamma_t = S_{(\lceil (1-\rho)N \rceil)}$.
Solve $\hat{v}_t = \frac{\sum_{i=1}^{n} I_{\{S(X_i) \geq \gamma_t\}} W(X_i; u, \hat{v}_{t-1}) X_{ij}}{\sum_{i=1}^{n} I_{\{S(X_i) \geq \gamma_t\}} W(X_i; u, \hat{v}_{t-1})}$ using the sample $X_1, \ldots X_n$, and set v_t accordingly.
Reiterate from step 3 until stopping condition is met.
Upon stopping: estimate $P(S(X_i) \geq \gamma) = \frac{1}{N} \sum_{i=1}^{N} I_{\{S(X_i) \geq \gamma\}} W(X_i; u, \hat{v}_t)$.

Algorithm C.2 Elitist CE algorithm (ECE).

Choose initial parameters N, ρ, and μ, and generate an initial parameter vector u.
Define $\hat{v}_0 = E^0 = u$. Set time tick $t = 1$.
Generate a random sample $X_1^t, \ldots, X_n^t$ from $\mathcal{X}$ restricted by $\hat{v}_{t-1}$.
Compute the performance $S(X_i^t)$ for every i and rank results $S_{(1)} \leq \ldots \leq S_{(n)}$.
Compute $\gamma^t = S_{(\lceil (1-\rho)N \rceil)}$. Find $E^t = \text{argmax}_{\hat{v}_{i=1,\ldots,t}} \gamma^i$.
For every hyperparameter v_j, set $\hat{v}_{t,j} = \frac{\sum_{i=1}^{n} I_{\{S(X_i^t) \geq \gamma^t\}} W(X_i^t; E^t) X_{ij}^t}{\sum_{i=1}^{n} I_{\{S(X_i^t) \geq \gamma^t\}} W(X_i^t; E^t)}$
STATE$t := t + 1$; reiterate from step 3 until stopping condition is met.

Algorithm C.3 Threshold estimation for decision value discretization.

Require: d_{min}, d_{max}, σ; D_{train} (decision values training data); P_-, P_+ (priors); ρ (resolution); σ (step size)
  $N_+ \Leftarrow 0; N_- \Leftarrow 0$
  $\lambda \Leftarrow \delta_{min}$
  while $\lambda < \delta_{max}$ **do**
    for each d in D_{train} **do**
      if $d < \lambda$ **then**
        $N_- \Leftarrow N_- + 1$
      else
        $N_+ \Leftarrow N_+ + 1$
      end if
    end for
    $N_+ \Leftarrow \frac{N_+}{|D_{train}|}$
    $N_- \Leftarrow \frac{N_-}{|D_{train}|}$
    if $\mid N_+ - P_+ \mid \leq \rho$ **and** $\mid N_- - P_- \mid \leq \rho$ **then**
      return λ
    end if
    $\lambda \Leftarrow \lambda + \sigma$
  end while
  Return λ

Algorithm C.4 EM algorithm for smoothing L1-normalized conditional probabilities.

Require: stopping criterion
  $\lambda \Leftarrow rand()$
  $k \Leftarrow 1$
  while stopping citerion not met **do**
    E-step:

$$\begin{aligned} E_{\lambda_k^c}(M_a \mid f_1^n) &= \textstyle\sum_i P(M_a \mid f_i) \\ &= \textstyle\sum_i \frac{P(M_a, f_i)}{\sum_i P(f_i)} \\ &= \textstyle\sum_i \sum_{c \in C} \frac{\lambda_k P_a(c|f_i^a)}{\lambda_k P_a(c|f_i^a) + (1-\lambda_k) P_b(c|f_i^b)} \end{aligned}$$

    M-step:

$$\lambda_{k+1} = \frac{E_{\lambda_k}(M_a|f_1^n)}{n}$$

    $k \Leftarrow k + 1$
  end while

Algorithm C.5 Compute the Gram error for two Gram matrices

Require: Two Gram matrices $G1, G2$.

1. Solve $G1 \cdot z = G2$
2. Compute $G1' = G1 \cdot z$
3. Compute the Gram error $E_G = \sum_{i=1}^{n} \sum_{j=1}^{n} (G1'_{ij} - G2_{ij})^2$

Algorithm C.6 Dimensionality reduction.

Require: Data matrix M; δ a distance measure defined on M; desired dimension m

Compute weight matrix W, diagonal weight matrix M, and graph Laplacian $L = M - W$, based on δ.
Solve $L\mathbf{v} = \lambda M\mathbf{v}$
Report m smallest non-zero eigenvectors.

Algorithm C.7 Normalize weights.

Require: M, a Laplacian Eigenmap; $\Lambda = \lambda_1 \ldots \lambda_c$ the computed eigenvalues

$\sigma \Leftarrow 0$
$\beta \Leftarrow 0$
$r \Leftarrow \#rows\ of M$
$c \Leftarrow \#columns\ of M$
for $j = 1 \ldots c$ **do**
  for $k = 1 \ldots r$ **do**
    $\sigma = \sigma + [M(k, j) \times \lambda_j]$
  end for
  $W_j \Leftarrow \frac{\sigma}{r}$
  $\sigma \Leftarrow 0$
end for

Algorithm C.8 Compute Eigenvalues.

Require: Data matrix M; δ a distance measure defined on M

Compute weight matrix W, diagonal weight matrix M, and graph Laplacian $L = M - W$, based on δ.
Solve $L\mathbf{v} = \lambda M\mathbf{v}$
Return all eigenvalues Λ

Algorithm C.9 Blockwise feature weighting.

Require: rs: row step size; cs: column step size; k; data M; δ_t a distance measure with $t \in \{NGD, EUCLID\}$.

$i \Leftarrow 0$
$r \Leftarrow \#rows\ of M$
$c \Leftarrow \#columns\ of M$
$W \Leftarrow [\ ]$
while $i \leq c$ **do**
  $\sigma \Leftarrow 0$
  $j \Leftarrow 0$
  while $j \leq r$ **do**
    $M_{sub} = M(j : j + rs, i : i + cs)$
    $\Lambda =$COMPUTE-EIGENVALUES(M_{sub}, δ_t)
    $W_i \ldots W_{i+cs} \Leftarrow W_i \ldots W_{i+cs}+$NORMALIZE-WEIGHTS$(M_{sub}, \Lambda)$
    $j \Leftarrow j + rs$
    $\sigma \Leftarrow \sigma + 1$
  end while
  $i \Leftarrow i + cs$
end while
$W \Leftarrow \frac{W}{\sigma}$

Algorithm C.10 The Hein-Maier algorithm for Manifold Denoising.

Require: Data $X = \{X_i\}_{i=1}^n \subseteq R^d$.

Let $\delta_{EUCLID}(X_i)$ be the Euclidean distance of X_i to its k-nearest neighbor.
Choose $\delta t, k$.
while Stopping criterion not satisfied **do**
  Compute the k-NN distances $\delta_{EUCLID}(X_i)$,$i = 1, \ldots, n$
  Compute the weights $w(X_i, X_j)$ of the graph with

$$w(X_i, X_i) = \quad 0$$

$$w(X_i, X_j) = \quad \exp\left(-\frac{\|X_i - X_j\|^2}{(\max\{\delta_{EUCLID}(X_i), \delta_{EUCLID}(X_j)\})^2}\right)$$

$$if \quad \| X_i - X_j \| \leq \max\{\delta_{EUCLID}(X_i), \delta_{EUCLID}(X_j)\}$$

  Compute the graph Laplacian $\Delta = 1 - D^{-1}W$
  Let X^α be the α-th component of vector X.
  Solve $X^\alpha(t+1) - X^\alpha(t) = -\delta t \Delta X^\alpha(t+1) \Rightarrow X^\alpha(t+1) = (1+\delta t\Delta)^{-1} X^\alpha(t)$, for $\alpha = 1, \ldots, d$.
end while

Algorithm C.11 A geodesic version of the Hein-Maier algorithm for Manifold Denoising.

Require: Data $X = \{X_i\}_{i=1}^n \subseteq R^d$.
Choose $\delta t, k$.
Let $\delta_{GD}(X_i)$ be the geodesic distance of X_i to its k-nearest neighbor.
while Stopping criterion not satisfied **do**
Compute the k-NN distances $\delta_{GD}(X_i)$,$i = 1, \ldots, n$
Compute the weights $w(X_i, X_j)$ of the graph with

$$w(X_i, X_i) = \quad 0$$

$$w(X_i, X_j) = \quad \exp\left(-\frac{K_{GD}(X_i,X_j)}{(\max\{\delta_{GD}(X_i),\delta_{GD}(X_j)\})}\right)$$

$$if\ K_{GD}(X_i, X_j) \leq \max\{\delta_{GD}(X_i), \delta_{GD}(X_j)\}$$

Compute the graph Laplacian $\Delta = 1 - D^{-1}W$
Let X^α be the α-th component of vector X.
Solve $X^\alpha(t+1) - X^\alpha(t) = -\delta t \Delta X^\alpha(t+1) \Rightarrow X^\alpha(t+1) = (1+\delta t\Delta)^{-1} X^\alpha(t)$, for $\alpha = 1, \ldots, d$.
end while

Algorithm C.12 Transductive learning through single method manifold denoising.

Require: γ; D_{train} (training data); D_{test} (test data); D'_{train} (development training data); D'_{test} (development test data); $\delta_t(\mathbf{x}, \mathbf{y})$ a distance function with $t \in \{GD, EUCLID\}$; $\mathcal{L}(f)$ an empirical loss function evaluating the function f.
Find $\arg\min_{\delta t,k}$ $\mathcal{L}$(MANIFOLD-DENOISE($D'_{train} \cup D'_{test}$;$\delta t, k, \delta_t$))
(K, Δ)=MANIFOLD-DENOISE($D_{train} \cup D_{test}$;$\delta t, k, \delta_t$)
for every test point $\mathbf{t}$ in D_{test} **do**
Let $NN(\mathbf{t})$ be the set of k nearest neighbors of $\mathbf{t}$ in K
if $NN(\mathbf{t})$ consists entirely of neighbors that are in the test set **then**
$Class(\mathbf{t}) \Leftarrow \arg\max_c \mid c \mid$ (majority class)
else
for every nearest neighbor $\mathbf{x} \in NN(\mathbf{t})$ with $Class(\mathbf{x}) = c$ such that $\mathbf{x} \in D_{train}$ **do**
$\delta \Leftarrow \delta_t(\mathbf{x}, \mathbf{t})$
Increment $v(c)$, the vote for c, with $\exp(-\gamma\delta)$
end for
$Class(\mathbf{t}) \Leftarrow \arg\max_c v(c)$
end if
end for

Algorithm C.13 Transductive learning through dual method manifold denoising.

Require: γ; D_{train} (training data); D_{test} (test data); D'_{train} (development training data); D'_{test} (development test data); $\delta_{GD \cup EUCLID}(\mathbf{x}, \mathbf{y})$ a distance function defined as $\frac{1}{2}[\delta_{GD}(\mathbf{x}, \mathbf{y}) + \delta_{EUCLID}(\mathbf{x}, \mathbf{y})]$; $\mathcal{L}(f)$ an empirical loss function evaluating the function f.
Find $\arg\min_{\delta t_1, k_1}$ $\mathcal{L}$(MANIFOLD-DENOISE($D'_{train} \cup D'_{test}$;$\delta t_1, k_1, \delta_{EUCLID}$))
Find $\arg\min_{\delta t_2, k_2}$ $\mathcal{L}$(MANIFOLD-DENOISE($D'_{train} \cup D'_{test}$;$\delta t_2, k_2, \delta_{GD}$))
$(K_{EUCLID}, \Delta_{EUCLID})$=
MANIFOLD-DENOISE($D_{train} \cup D_{test}$;$\delta t_1, k_1, \delta_{EUCLID}$)
(K_{GD}, Δ_{GD})=MANIFOLD-DENOISE($D_{train} \cup D_{test}$;$\delta t_2, k_2, \delta_{GD}$)
for every test point $\mathbf{t}$ in D_{test} **do**
  Let $NN(\mathbf{t})$ be the set of $k_1 + k_2$ nearest neighbors of $\mathbf{t}$ in $K_{GD} \cup K_{EUCLID}$
  if $NN(\mathbf{t})$ consists entirely of neighbors that are in the test set **then**
    $Class(\mathbf{t}) \Leftarrow \arg\max_c \mid c \mid$ (majority class)
  else
    for every nearest neighbor $\mathbf{x} \in NN(\mathbf{t})$ with $Class(\mathbf{x}) = c$ such that $\mathbf{x} \in D_{train}$ **do**
      $\delta \Leftarrow \delta_{GD \cup EUCLID}(\mathbf{x}, \mathbf{t})$
      Increment $v(c)$, the vote for c, with $\exp(-\gamma\delta)$
    end for
    $Class(\mathbf{t}) \Leftarrow \arg\max_c v(c)$
  end if
end for

Algorithm C.14 Accuracy optimization of yield.

Require: Training data Tr; test data Te; development data D_1, D_2; stepsize σ; a first-level classifier C_1 and a meta-classifier C_2; a loss function $\mathcal{L}$ (accuracy); pre-specified maximum loss λ (minimum accuracy).
Train first-level classifier on training part of D_1; apply to test part of D_1.
Determine Platt mapping P from decision values to posterior class probabilities on the basis of output of the previous step.
Using a two-parameter sweep with stepsize σ, find $\text{argmax}_{\theta_l, \theta_u}$ $\mathcal{L}(C_2(D_2; \theta_l, \theta_u)) \approx \lambda$.
Train C_1 on Tr.
Apply C_1 to Te.
Map decision values produced in the previous step to Platt scores using P.
Apply C_2 to the output of the previous step and threshold according to θ_l, θ_u.

Algorithm C.15 Obtaining thresholds θ_l, θ_u from development data, and optimizing for accuracy ($\hat{x}$ denotes the true class of x).

Require: K; D_{dev} (development data); σ (stepsize);$\delta_t(\mathbf{x}, \mathbf{y})$ a distance function with $t \in \{NGD, EUCLID\}$; γ a weighting parameter; $\theta_{min}, \theta_{max}$
Split D_{dev} into training/test partitions $D_{dev.train}$ and $D_{dev.test}$
$M_{NGD} \Leftarrow$ train SVM (NGD) on $D_{dev.train}$
$M_{EUCLID} \Leftarrow$ train SVM (linear kernel) on $D_{dev.train}$
For every test point in $D_{dev.test}$, find its K nearest Euclidean neighbors in M_{EUCLID} using $\delta_{EUCLID}(\mathbf{x}, \mathbf{y})$ (NN_{EUCLID}).
For every test point in $D_{dev.test}$, find its K nearest geodesic neighbors in M_{NGD} using $\delta_{NGD}(\mathbf{x}, \mathbf{y})$ (NN_{NGD}).
$\theta_l \Leftarrow \theta_{min}$
while $\theta_l \leq \theta_{max}$ **do**
  $\theta_u \Leftarrow \theta_{max}$
  while $\theta_u \geq \theta_l$ **do**
    for all datapoints x in $D_{dev.test}$ **do**
      for every class c **do**
        $Vote_c \Leftarrow 0$
      end for
      if $\overline{H}(NN_{NGD}(x)) \leq \theta_l$ **then**
        for all datapoints y in $NN_{NGD}(x)$ **do**
          $c \Leftarrow \hat{y}$
          $Vote_c \Leftarrow Vote_c + e^{-\gamma \delta_{NGD}(\mathbf{x},\mathbf{y})}$
        end for
      else
        if $\overline{H}(NN_{NGD}(x)) \geq \theta_u$ **then**
          for all datapoints y in $NN_{EUCLID}(x)$ **do**
            $c \Leftarrow \hat{y}$
            $Vote_c \Leftarrow Vote_c + e^{-\gamma \delta_{EUCLID}(\mathbf{x},\mathbf{y})}$
          end for
        else
          if $\theta_l < \overline{H}(NN_{NGD}(x)) < \theta_u$ **then**
            for all datapoints y in $NN_{NGD}(x)$ **do**
              $c \Leftarrow \hat{y}$
              $Vote_c \Leftarrow Vote_c + e^{-\gamma \delta_{EUCLID}(\mathbf{x},\mathbf{y})} + e^{-\gamma \delta_{NGD}(\mathbf{x},\mathbf{y})}$
            end for
          end if
        end if
      end if
      $C \Leftarrow \arg\max_c Vote_c$
      if $C = \hat{x}$ **then**
        $C \Leftarrow C + 1$
      else
        $E \Leftarrow E + 1$
      end if
    end for
    $\theta_l \Leftarrow \theta_l + \sigma$
  end while
  $\theta_u \Leftarrow \theta_u - \sigma$
end while
Return: θ_l, θ_u and accuracy=$\frac{C}{C+E}$

Appendix D

Quantum interpretation

Quantum information science (e.g. Mermin [2007], and van Rijsbergen [2004]) is an exciting, rapidly expanding field that studies the design of algorithms amenable to the architecture of quantum computers. While large-scale quantum computing is not a reality yet, quantum information science is generally accepted as an evolutionary next step in information science, where information processing becomes a physical process subject to quantum laws. Quantum computing opens up fresh perspectives on complexity, with new time-space complexity classes such as BQP, *bounded error quantum polynomial time*. It is assumed that BQP is a strict superset of P, the class of problems that are solvable by a deterministic Turing Machine in polynomial time; see Watrous [2009]). Several sub-exponential and polynomial quantum algorithms for difficult problems have been proposed, e.g. Beals et al. [1998], Kuperberg [2005], and the famous Shor [1997], who presents an algorithm for polynomial-time prime factorization. Grover [1996] is another example of a fast quantum algorithm for database search with favorable time complexity, compared to its classical analogue. The latter two algorithm form the basis of many quantum algorithms that were subsequently proposed,

Central to any quantum algorithm is the clever exploitation of *superposition* and *entanglement* of information. Quantum computers use *qubits* or quantum bits, units of information that unlike classical bits can be in three representational states at a time: they can be on, off, or both. Classical bits, on the other hand, are always either on or off. A computer handling qubits can therefore handle much more information at a certain time compared to a classical computer: for n bits, a classical computer is in one of the 2^n possible states, whereas a quantum computer is in all of these states simultaneously. The superposition of qubits leads to a description of qubits as a vector of *amplitudes*: the chance that the qubit ψ is in one of its classical states 0 or 1 when we measure it (the

following notation is called the *ket* notation (Mermin [2007]):

$$|\psi\rangle = \alpha_0 |0\rangle + \alpha_1 |1\rangle = \begin{pmatrix} \alpha_0 \\ \alpha_1 \end{pmatrix} \tag{D.1}$$

Measurement produces a classical state $|i\rangle$, with probability $\mid \alpha_i \mid^2$. The squared amplitudes form a probability distribution, hence

$$\sum_i \mid \alpha_i \mid^2 = 1 \tag{D.2}$$

Entanglement is the phenomenon that two objects in a quantum setting are correlated in the sense that measurement of the one directly determines the state of the other. This correlation can be non-local.

One of the interpretations of quantum mechanics (the *Kopenhagen* interpretation; e.g. Jammer [1966]) is that measurement of a quantum object automatically destroys quantum superposition, and reduces the quantum object to a classical object. This phenomenon has been described empirically by the *double slit* experiments (e.g. Mittelstaedt et al. [1987]).

The connection between quantum information science and machine learning has been investigated in a number of publications. Ventura and Martinez [2000] propose a quantum-theoretic formulation of an associative memory, a neural network architecture with content-addressable memory (i.e. memory is accessed through matching operations of queries with stored samples, as opposed to index-based memory access). Their quantum network has exponentially more memory than a classical Hopfield network. Quantum neural networks are more thoroughly investigated in Ezhov and Berman [2003]. Ricks and Ventura [2003] address the issue of training procedure for quantum multilayer perceptrons. Aïmeur et al. [2006] and Aïmeur et al. [2007] present a quantum-theoretic reconstruction of divisive and $k-medians$ clustering. The perspective offered by quantum computing to information science consists of highly efficient, massively parallel algorithms that will lead to new possibilities for complex data analysis.

In this appendix we present a quantum-theoretic interpretation of the hybrid representation of document geometry as presented in Chapter 8. As it turns out, also the multinomial simplex that arises from L1-normalization is interpretable in a quantum context. These interpretations may be viewed as a necessary pre-condition for quantum algorithms.

D.1 Superposition of document geometry

The hybrid perspective we outlined so far on document geometry allows for a quantum information-theoretic interpretation when we assume a certain test point x is superimposed in three classical states until measurement occurs. Here, the three classical states of x are the states defined by the thresholds θ_l and θ_u:

the first state arises when $\overline{H}(NN(x)) \leq \theta_l$, the second when $\overline{H}(NN(x)) \geq \theta_u$, and the third when $\theta_l < \overline{H}(NN(x)) < \theta_u$. A test point is in either one of these states with a state-dependent probability or amplitude. Let $|01\rangle$ denote the classical state of 'x having a nearest neighborhood with curved geometry', and $|10\rangle$ the classical state of 'x having a nearest neighborhood with flat geometry'. Let $|11\rangle$ denote the classical state of 'x having a nearest neighborhood with mixed geometry'; this state can be derived from the other two states, as we will outline. The zero state $|00\rangle$ is the non-existent state where a document is neither in one of the three other states.

A state ψ of a certain test point x can be described as

$$|\psi\rangle = \alpha_{00}\,|00\rangle + \alpha_{01}\,|01\rangle = \alpha_{10}\,|01\rangle + \alpha_{11}\,|11\rangle \tag{D.3}$$

subject to the normalization constraint

$$\sum_i \mid \alpha_i \mid^2 = 1 \tag{D.4}$$

and treating $\alpha_{00} = 0$.

Let

$$\begin{aligned} H_{min} &= min\{\overline{H}(NN(x_i))\} \\ H_{max} &= max\{\overline{H}(NN(x_i))\} \end{aligned} \tag{D.5}$$

Further

$$f_{\theta_l}(x) = \begin{cases} \frac{1}{\theta_l - H_{min}} \; for \; H_{min} \leq \overline{H}(NN(x)) \leq \theta_l \\ 0 \; for \; x < H_{min} \; or \; x > \theta_l \end{cases} \tag{D.6}$$

$$f_{\theta_u}(x) = \begin{cases} \frac{1}{H_{max} - \theta_u} \; for \; \theta_u \leq \overline{H}(NN(x)) \leq H_{max} \\ 0 \; for \; x < \theta_u \; or \; x > H_{max} \end{cases} \tag{D.7}$$

with x a test point. We have

$$\begin{aligned} f_{\theta_l}(x) \geq 0 \; \forall x \wedge \textstyle\int_{-\inf}^{\inf} f_{\theta_l}(x) = 1 \\ f_{\theta_u}(x) \geq 0 \; \forall x \wedge \textstyle\int_{-\inf}^{\inf} f_{\theta_u}(x) = 1 \end{aligned} \tag{D.8}$$

This implies these functions are probability density functions. We then obtain the following probabilities that describe the chance of being in respectively the curved and Euclidean state:

$$\begin{aligned} P(\theta_l) &= \textstyle\int_{H_{min}}^{\theta_l} f_{\theta_l}(x)dx \\ P(\theta_u) &= \textstyle\int_{\theta_u}^{H_{max}} f_{\theta_u}(x)dx \end{aligned} \tag{D.9}$$

Notice that the probability of being in the third state, both Euclidean and curved geometry (which means the average entropy of the neighborhood of point x is in between θ_k and θ_u) can be obtained as

$$P(\theta_l \cdot \theta_u) = 1 - P(\theta_l) - P(\theta_u) \tag{D.10}$$

E.g., when θ_l and θ_u coalesce, there is no mixed region, and the probability of ending up there is zero ($P(\theta_l) + P(\theta_u) = 1$). This means we are viewing the thresholds θ_l and θ_u as *amplitudes of classical states*, and that a chance of observing the third state arises when $\theta_l \neq \theta_u$.

The superposition state Ψ can then be described as

$$|\Psi\rangle \mapsto 0\,|00\rangle + \sqrt{P(\theta_l)}\,|01\rangle + \sqrt{P(\theta_u)}\,|10\rangle + \sqrt{P(\theta_l \cdot \theta_u)}\,|11\rangle \tag{D.11}$$

This can be expressed in terms of a quantum gate, which we specify by its matrix operation:

$$\mathcal{R} : \begin{pmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{pmatrix} \mapsto \begin{pmatrix} 0 & 0 & 0 & 0 \\ 0 & \sqrt{P(\theta_l)} & 0 & 0 \\ 0 & 0 & \sqrt{P(\theta_u)} & 0 \\ 0 & 0 & 0 & \sqrt{P(\theta_l \cdot \theta_u)} \end{pmatrix} \tag{D.12}$$

and the following quantum circuit can be drawn (where M is measurement, producing classical information)

$$|0\rangle \quad \boxed{\mathcal{R}} \;\boxed{M} \tag{D.13}$$

Applying the gate $\mathcal{R}$ to state $|\Psi\rangle$ we obtain

$$\sqrt{P(\theta_l)}\,|01\rangle + \sqrt{P(\theta_u)}\,|10\rangle + \sqrt{P(\theta_l \cdot \theta_u)}\,|11\rangle \tag{D.14}$$

Subsequent measurement produces either state 01, 10, or the mixed state 11 with respective probabilities

$$\begin{aligned} |\sqrt{P(\theta_l)}|^2 &= P(\theta_l) \\ |\sqrt{P(\theta_u)}|^2 &= P(\theta_u) \end{aligned} \tag{D.15}$$

and, in case $P(\theta_l) + P(\theta_u) < 1$

$$1 - P(\theta_l) - P(\theta_u) = P(\theta_l \cdot \theta_u) \tag{D.16}$$

D.2 A word drawing game

We can in fact give a quantum interpretation to the multinomial simplex as well. Given a vocabulary V of n terms, a given document D of k words is an

instantiation of V, representable as a bit sequence of n bits, where every bit b_i represents either that the word w_i is in document D ($b_i = 1$) or it is not ($b_i = 0$). Document space, the set of all possible instantiations of V, can then be interpreted as the superposition of the n terms in V as both *on* and *off*:

$$|\Psi\rangle = \alpha_1 |\Psi_1\rangle + \ldots + \alpha_{2^n} |\Psi_{2^n}\rangle \tag{D.17}$$

where Ψ is the (huge) superposition of the 2^n possible instantiations of the lexicon[1].

Similarly, L1-normalized document representations can be seen as superposed outcomes of a *word drawing game*. Given a document D of n words, a quantum version of the L1-representation is

$$|w_1 \ldots w_n\rangle = \sqrt{\frac{\mid w_1 \mid}{\sum_i \mid w_i \mid}} |w_1\rangle + \ldots + \sqrt{\frac{\mid w_n \mid}{\sum_i \mid w_i \mid}} |w_n\rangle \tag{D.18}$$

with

$$|w_i\rangle = 1 \tag{D.19}$$

for every word w_i in D. For instance, given a vocabulary $V = \{a, b, c\}$, and a document $D = \{a, a, b, c, c\}$, we have

$$|abc\rangle = \sqrt{\frac{2}{5}} |a\rangle + \sqrt{\frac{1}{5}} |b\rangle + \sqrt{\frac{2}{5}} |c\rangle \tag{D.20}$$

and since

$$\begin{aligned} \alpha_1 &= \sqrt{\tfrac{2}{5}} \\ \alpha_2 &= \sqrt{\tfrac{1}{5}} \\ \alpha_3 &= \sqrt{\tfrac{2}{5}} \end{aligned} \tag{D.21}$$

we have

$$\sum_i \mid \alpha_i \mid^2 = 1 \tag{D.22}$$

The superposed state $|abc\rangle$ describes the simultaneous probability of drawing words a, b, c, and a classical measurement would obtain either a, b, or c, with a probability equal to its L1-normalized frequency. Therefore, in a quantum context, the L1-representation of documents can be interpreted as superposition of all states of n words, with the L1-normalized frequencies the amplitudes of the classical states: the *prior* probability of observing a word w given a document D, $p(w \mid D)$.

[1] Notice that this holds for the view of documents as bags of words, i.e. unordered sets with repetition. The space of all permutation variants –of which only a fraction is well-formed– is much bigger: $2^n!$ permutations

The reconstruction in terms of quantum data structures of both the calibration technique proposed in Chapter 8 and the multinomial simplex opens up possibilities for the construction of quantum algorithms, a topic that is beyond the scope of this thesis, but which, given the increasing awareness of quantum information science in the field of machine learning, will undoubtedly gain importance in the next years.

Summary

For quite some time, learning systems have been applied to the analysis of language. learning systems, part of the discipline of *machine learning*, learn to discriminate between objects of different classes, on the basis of examples called *training data*. Machine learning methods can be divided into roughly two types: memory-based methods, storing all examples in memory, and comparing new cases with these stored examples, and model-based methods, that concisely represent the set of examples in a succinct model in which, for instance, only the boundary cases representing the separation between classes are stored. A learning system is trained on its training data, after which it is capable of labeling – with a variable degree of success – new (*test*) cases with a classification. A trained learning system is called a *classifier*.

During both the learning stage and classification stage, learning systems deploy similarity measures. For instance, a memory-based system will measure the similarity of a test case with cases stored in memory. The set of stored cases that most closely resemble the test case determine the classification of the latter. This type of similarity can be expressed as a distance measure, according to which the data can be described through a vocabulary of properties (*features*) as a *feature space*. Every feature space comes equipped with corresponding distance measures that befit the intrinsic geometry of the space.

For the application of learning systems to the analysis of language, the so-called *vector space* is traditionally used for representations. In this space, linguistic objects such as texts are represented as vectors: points in a high-dimensional space. Characteristic of this space is the fact that it possesses a flat structure, in which distances are measured along straight lines. From a two-dimensional perspective, one can imagine this space as the space spanned by linear functions of the form $y = ax + b$. The vector space model has proved to be quite successful for *document retrieval* (where distance plays a role in the process of finding documents that match a certain query) and machine learning. The corresponding distance measures are called *Euclidean* distance measures.

Recently, the set of distance measures for the application of learning systems

to language has been extended with measures that assume a curved space: the so-called *geodesic* distance measures. These distance measures treat local distances as Euclidean, and measure global distance along curved lines. This can be compared to the practice of measuring distance on our globe: the curvature of the globe does not come into play when measuring the distance between two objects that are close to each other, but does play a role when measuring the distance between objects that are thousands of miles apart. Documents can be embedded into a geodesic space, through a simple transformation based on normalized frequencies of e.g. words. For effective use of the resulting *probability distributions*, it is necessary that documents possess a certain length; otherwise, the notion of frequency would be meaningless. As it turns out, geodesic distance measures yield highly accurate classifiers.

In this thesis, we examine the conditions under which geodesic classifiers perform optimally. First, we carry out a number of experiments in which we assess the accuracy of these classifiers. We subsequently propose to extend the standard technology with two new facilities:

- A method to apply geodesic classifiers to very short texts, with an eventual sequential relation between the constituting parts. Examples are certain feature-based learning tasks with a limited length, such as the prediction of the attachment of prepositional phrases to preceding verbs or nouns, or the prediction of the diminutive suffix of a noun on the basis of a limited number of phonetic properties its final syllables.

- A method to combine heterogeneous information (such as frequencies of separate words and frequencies of word combinations) within one classifier.

We demonstrate that the proposed modifications boost classifier performance, and that they correspond with standard formal operations on a geodesic space.

Next, we investigate in detail whether document data embedded in a geodesic space should be uniformly analyzed with geodesic distance measures. We show analytically that the inverse cosine, a crucial ingredient of geodesic distance measures, displays weak performance on some regions in its spectrum. We relate this phenomenon to the notion of *entropy*: flat probability distributions of embedded data lead to suboptimal performance of geodesic classifiers. On the basis of two empirical studies, we explain this observation from an inverse relationship between entropy and curvature: the higher the entropy of certain data, the lower the amount of curvature representing this data. Our conclusion is that textual data embedded in a geodesic space should not be uniformly analyzed with geodesic methods.

Subsequently, we propose a calibration technique for classifiers. This technique, based on the estimation on two thresholds on the class probabilities emitted by a classifier, allows the classifier to factor out hard cases it cannot classify with a pre-specified accuracy. We generalize this technique to a method

with which we can train a classifier to switch from geodesic distance measures back to Euclidean distance measures, depending on the entropy of the local neighborhood of test data. In this way, we create a *back-off* system that under certain conditions backs off from a more complex geodesic distance measure to a less complex Euclidean distance measure. We demonstrate the benefits of this method as compared to uncalibrated methods. In addition, we propose an alternative distance measure based on a method from cartography that is specifically well-suited for high-entropy data.

Samenvatting

Al geruime tijd worden lerende systemen toegepast op de analyse van taal. Lerende systemen, ressorterend onder het vakgebied van de *machine learning*, leren het onderscheid tussen objecten, verdeeld in klassen, op basis van voorbeelden (zogenaamde *training data*). Daarbij zijn er ruwweg twee soorten methoden: geheugengebaseerde methoden, die alle voorbeelden opslaan in het geheugen, en nieuwe gevallen vergelijken met die opgeslagen gevallen, en modelgebaseerde systemen, die de verzameling voorbeelden beknopt samenvatten in een gecondenseerd model waarin bijvoorbeeld alleen de grensgevallen – die de scheiding tussen klassen markeren – worden opgenomen. Een lerend systeem dient te worden getraind, waarna het in staat is met meer of minder succes nieuwe (*test*) data te voorzien van een classificatie. Een getraind lerend systeem wordt een *classifier* genoemd.

Lerende systemen maken bij het leren, en bij het toepassen van het geleerde, gebruik van gelijkenismaten. Zo zal een geheugengebaseerd systeem voor het bepalen van de klasse van een nieuw testgeval kijken naar de gelijkenis van dat geval met opgeslagen voorbeelden. De voorbeelden die in de training data het meest lijken op het testgeval bepalen de classificatie van de laatste. Dit soort gelijkenis kan worden uitgedrukt als een afstandsmaat, waarbij de data kan worden beschreven via een vocabulaire van eigenschappen (*features*) als een *feature-ruimte*. Bij elke feature-ruimte horen afstandsmaten die passen bij de specifieke structuur van de ruimte.

Bij de toepassing van lerende systemen op de analyse van taal wordt traditioneel gebruik gemaakt van een zogeheten *vectorruimte*. In deze ruimte worden talige objecten zoals teksten gerepresenteerd als vectoren: punten in een hoogdimensionale ruimte. Kenmerkend aan deze ruimte is dat zij een vlakke structuur heeft, en dat afstanden worden gemeten langs rechte lijnen. Tweedimensionaal is deze ruimte voor te stellen als de ruimte opgespannen door lineaire functies van de vorm $y = ax + b$. Het vector space model is zeer succesvol gebleken in de wereld van de *document retrieval*, (waar afstand een rol speelt bij het vinden van documenten die bij een zoekvraag passen) en de machine learning. De

bijbehorende afstandsmaten worden *Euclidische* afstandsmaten genoemd.

Recentelijk is de verzameling afstandsmaten voor de toepassing van lerende systemen op taal uitgebreid met maten die uitgaan van een gekromde ruimte: de zogeheten *geodetische* afstandsmaten. Deze afstandsmaten behandelen strikt lokale afstanden als Euclidisch, en meten globalere afstanden langs gekromde lijnen. Dit is te vergelijken met de manier waarop men op onze globe afstanden meet: voor het meten van de afstand tussen twee objecten op geringe afstand van elkaar speelt de kromming van de globe geen rol, maar wel als die twee objecten enkele duizenden kilometers van elkaar verwijderd zijn. Documenten kunnen via een eenvoudige, op genormaliseerde (e.g. woord-)frequenties gebaseerde transformatie worden 'ingebed' in een geodetische ruimte. Voor een effectief gebruik van deze aldus ontstane *waarschijnlijkheidsdistributies* is het nodig dat de teksten een zekere lengte hebben, omdat anders de notie frequentie zinloos zou zijn. Gebleken is dat geodetische afstandsmaten aanleiding geven tot zeer nauwkeurige classifiers.

In dit proefschrift onderzoeken we de condities waaronder geodetische classifiers optimaal presteren. Allereerst voeren we een aantal experimenten uit waarin we de nauwkeurigheid van deze classifiers evalueren. Vervolgens breiden we de standaardtechnologie uit met twee nieuwe faciliteiten:

- Een methode om geodetische classifiers toe te passen op zeer korte teksten, met een eventueel sequentiële verhouding tussen de samenstellende delen. Hierbij moet worden gedacht aan feature-gebaseerde leertaken met een beperkte lengte, zoals het voorspellen van de aanhechting van prepositionele frases aan voorafgaande zelfstandige naamwoorden of werkwoorden, of aan het voorspellen van het verkleinsuffix van een zelfstandig naamwoord op basis van een beperkt aantal fonetische eigenschappen van zijn laatste lettergrepen.

- Een methode om heterogene informatie (zoals frequenties van losse woorden en frequenties van woordcombinaties) te combineren binnen één classifier.

We laten zien dat de voorgestelde modificaties tot hogere prestaties van de classifiers leiden, en dat ze corresponderen met standaard formele operaties op een geodetische dataruimte.

Vervolgens onderzoeken we in detail of document data ingebed in een geodetische ruimte uniform met geodetische afstandsmaten dient te worden benaderd. We laten langs analytische weg zien dat de inverse cosinus, een cruciaal ingredient van geodetische afstandsmaten, op bepaalde plekken in zijn spectrum zwakke prestaties levert. We relateren dit verschijnsel aan de notie van *entropie*: vlakke waarschijnlijkheidsdistributies van ingebedde data leiden tot suboptimale prestaties van geodetische classifiers. Deze observatie verklaren we via twee empirische studies vanuit een invers verband tussen entropie en kromming: hoe hoger de entropie van bepaalde data, hoe lager de kromming van de

ruimte die de data representeert. Onze conclusie is dat tekstdata ingebed in een geodetische ruimte niet uniform kan worden geanalyseerd met geodetische methodes.

Vervolgens stellen we een calibratietechniek voor classifiers voor. Deze techniek, gebaseerd op het schatten van twee drempelwaarden op de klassekansen van een classifier, stelt een classifier in staat om zelf moeilijke gevallen te identificeren die niet met een vooraf gespecificeerde nauwkeurigheid kunnen worden geclassificeerd. De calibratietechniek generaliseren we vervolgens tot een methode waarmee we een classifier kunnen trainen op het zelf schakelen tussen geodetische en Euclidische afstandsmaten, al naar gelang de entropie van de lokale omgeving van test data hoger of lager is. Op deze manier creëren we een *back-off* systeem dat onder bepaalde condities terugschakelt van een complexere geodetische afstandsmaat naar een minder complexe Euclidische afstandsmaat. We demonstreren dat deze methode tot betere resultaten leidt dan ongecalibreerde methoden. Ook stellen we een alternatieve afstandsmaat voor gebaseerd op een methode uit de cartografie die met name geschikt is voor hoog-entropische data.

Curriculum Vitae

Stephan Alexander Raaijmakers was born in Hazerswoude Rijndijk. After attending secondary school (the Christelijk Lyceum in Alphen aan den Rijn), he obtained his "propedeuse" at Leiden University in Dutch Language and Literature, before moving on to Computational Linguistics (part of General Linguistics) with minors in Information Science. As a student he was elected to implement the Dutch version of the world's first commercial natural language database interface: Q&A, produced by Symantec, localized by Transmodul GmbH, Saarbrücken. During his study, he specialized in formal logic (theorem proving with substructural logics, typed lambda calculus, dynamic logic) and deterministic computational models for parsing natural language. After graduation, he worked as a researcher at the Institute for Language Technology and Artificial Intelligence (ITK) of Tilburg University on a variety of computational and semantic issues, including proof-theoretic approaches to parsing, and the dynamic-semantic analysis of ellipsis. Subsequently, he was employed as a computational linguist at the Institute for Dutch Lexicology (INL) in Leiden, addressing large-scale corpus annotation with statistical and machine learning models. Since 2000, he is employed as a researcher at TNO, specializing in the application of machine learning techniques to a wide variety of topics, ranging from text analytics to image analysis. He is an active journal and conference paper reviewer, and is involved as a programme committee member in the organization of a number of conferences.

SIKS Dissertation Series

1998 [1]

1. Johan van den Akker (CWI) *DEGAS - An Active, Temporal Database of Autonomous Objects*

2. Floris Wiesman (UM) *Information Retrieval by Graphically Browsing Meta-Information*

3. Ans Steuten (TUD) *A Contribution to the Linguistic Analysis of Business Conversations within the Language/Action Perspective*

4. Dennis Breuker (UM) *Memory versus Search in Games*

5. E.W.Oskamp (RUL) *Computerondersteuning bij Straftoemeting*

1999

1. Mark Sloof (VU) *Physiology of Quality Change Modelling; Automated modelling of Quality Change of Agricultural Products*

2. Rob Potharst (EUR) *Classification using decision trees and neural nets*

3. Don Beal (UM) *The Nature of Minimax Search*

4. Jacques Penders (UM) *The practical Art of Moving Physical Objects*

[1] Abbreviations: SIKS – Dutch Research School for Information and Knowledge Systems; CWI– Centrum voor Wiskunde en Informatica, Amsterdam; EUR – Erasmus Universiteit, Rotterdam; KUB– Katholieke Universiteit Brabant, Tilburg; KUN – Katholieke Universiteit Nijmegen; OU – Open Universiteit; RUG – Rijksuniversiteit Groningen; RUL – Rijksuniversiteit Leiden; RUN – Radboud Universiteit Nijmegen; TUD – Technische Universiteit Delft; TU/e – Technische Universiteit Eindhoven; UL – Universiteit Leiden; UM – Universiteit Maastricht; UT – Universiteit Twente, Enschede; UU – Universiteit Utrecht; UvA – Universiteit van Amsterdam; UvT – Universiteit van Tilburg; VU – Vrije Universiteit, Amsterdam.

5. Aldo de Moor (KUB) *Empowering Communities: A Method for the Legitimate User-Driven Specification of Network Information Systems*
6. Niek J.E. Wijngaards (VU) *Re-design of compositional systems*
7. David Spelt (UT) *Verification support for object database design*
8. Jacques H.J. Lenting (UM) *Informed Gambling: Conception and Analysis of a Multi-Agent Mechanism for Discrete Reallocation.*

2000

1. Frank Niessink (VU) *Perspectives on Improving Software Maintenance*
2. Koen Holtman (TU/e) *Prototyping of CMS Storage Management*
3. Carolien M.T. Metselaar (UvA) *Sociaal-organisatorische gevolgen van kennistechnologie; een procesbenadering en actorperspectief.*
4. Geert de Haan (VU) *ETAG, A Formal Model of Competence Knowledge for User Interface Design*
5. Ruud van der Pol (UM) *Knowledge-based Query Formulation in Information Retrieval.*
6. Rogier van Eijk (UU) *Programming Languages for Agent Communication*
7. Niels Peek (UU) *Decision-theoretic Planning of Clinical Patient Management*
8. Veerle Coup (EUR) *Sensitivity Analyis of Decision-Theoretic Networks*
9. Florian Waas (CWI) *Principles of Probabilistic Query Optimization*
10. Niels Nes (CWI) *Image Database Management System Design Considerations, Algorithms and Architecture*
11. Jonas Karlsson (CWI) *Scalable Distributed Data Structures for Database Management*

2001

1. Silja Renooij (UU) *Qualitative Approaches to Quantifying Probabilistic Networks*
2. Koen Hindriks (UU) *Agent Programming Languages: Programming with Mental Models*
3. Maarten van Someren (UvA) *Learning as problem solving*

4. Evgueni Smirnov (UM) *Conjunctive and Disjunctive Version Spaces with Instance-Based Boundary Sets*
5. Jacco van Ossenbruggen (VU) *Processing Structured Hypermedia: A Matter of Style*
6. Martijn van Welie (VU) *Task-based User Interface Design*
7. Bastiaan Schonhage (VU) *Diva: Architectural Perspectives on Information Visualization*
8. Pascal van Eck (VU) *A Compositional Semantic Structure for Multi-Agent Systems Dynamics.*
9. Pieter Jan 't Hoen (RUL) *Towards Distributed Development of Large Object-Oriented Models, Views of Packages as Classes*
10. Maarten Sierhuis (UvA) *Modeling and Simulating Work Practice BRAHMS: a multiagent modeling and simulation language for work practice analysis and design*
11. Tom M. van Engers (VU) *Knowledge Management: The Role of Mental Models in Business Systems Design*

2002

1. Nico Lassing (VU) *Architecture-Level Modifiability Analysis*
2. Roelof van Zwol (UT) *Modelling and searching web-based document collections*
3. Henk Ernst Blok (UT) *Database Optimization Aspects for Information Retrieval*
4. Juan Roberto Castelo Valdueza (UU) *The Discrete Acyclic Digraph Markov Model in Data Mining*
5. Radu Serban (VU) *The Private Cyberspace Modeling Electronic Environments inhabited by Privacy-concerned Agents*
6. Laurens Mommers (UL) *Applied legal epistemology; Building a knowledge-based ontology of the legal domain*
7. Peter Boncz (CWI) *Monet: A Next-Generation DBMS Kernel For Query-Intensive Applications*
8. Jaap Gordijn (VU) *Value Based Requirements Engineering: Exploring Innovative E-Commerce Ideas*

9. Willem-Jan van den Heuvel(KUB) *Integrating Modern Business Applications with Objectified Legacy Systems*

10. Brian Sheppard (UM) *Towards Perfect Play of Scrabble*

11. Wouter C.A. Wijngaards (VU) *Agent Based Modelling of Dynamics: Biological and Organisational Applications*

12. Albrecht Schmidt (Uva) *Processing XML in Database Systems*

13. Hongjing Wu (TU/e) *A Reference Architecture for Adaptive Hypermedia Applications*

14. Wieke de Vries (UU) *Agent Interaction: Abstract Approaches to Modelling, Programming and Verifying Multi-Agent Systems*

15. Rik Eshuis (UT) *Semantics and Verification of UML Activity Diagrams for Workflow Modelling*

16. Pieter van Langen (VU) *The Anatomy of Design: Foundations, Models and Applications*

17. Stefan Manegold (UvA) *Understanding, Modeling, and Improving Main-Memory Database Performance*

2003

1. Heiner Stuckenschmidt (VU) *Ontology-Based Information Sharing in Weakly Structured Environments*

2. Jan Broersen (VU) *Modal Action Logics for Reasoning About Reactive Systems*

3. Martijn Schuemie (TUD) *Human-Computer Interaction and Presence in Virtual Reality Exposure Therapy*

4. Milan Petković (UT) *Content-Based Video Retrieval Supported by Database Technology*

5. Jos Lehmann (UvA) *Causation in Artificial Intelligence and Law - A modelling approach*

6. Boris van Schooten (UT) *Development and specification of virtual environments*

7. Machiel Jansen (UvA) *Formal Explorations of Knowledge Intensive Tasks*

8. Yongping Ran (UM) *Repair Based Scheduling*

9. Rens Kortmann (UM) *The resolution of visually guided behaviour*

10. Andreas Lincke (UvT) *Electronic Business Negotiation: Some experimental studies on the interaction between medium, innovation context and culture*

11. Simon Keizer (UT) *Reasoning under Uncertainty in Natural Language Dialogue using Bayesian Networks*

12. Roeland Ordelman (UT) *Dutch speech recognition in multimedia information retrieval*

13. Jeroen Donkers (UM) *Nosce Hostem - Searching with Opponent Models*

14. Stijn Hoppenbrouwers (KUN) *Freezing Language: Conceptualisation Processes across ICT-Supported Organisations*

15. Mathijs de Weerdt (TUD) *Plan Merging in Multi-Agent Systems*

16. Menzo Windhouwer (CWI) *Feature Grammar Systems - Incremental Maintenance of Indexes to Digital Media Warehouses*

17. David Jansen (UT) *Extensions of Statecharts with Probability, Time, and Stochastic Timing*

18. Levente Kocsis (UM) *Learning Search Decisions*

2004

1. Virginia Dignum (UU) *A Model for Organizational Interaction: Based on Agents, Founded in Logic*

2. Lai Xu (UvT) *Monitoring Multi-party Contracts for E-business*

3. Perry Groot (VU) *A Theoretical and Empirical Analysis of Approximation in Symbolic Problem Solving*

4. Chris van Aart (UvA) *Organizational Principles for Multi-Agent Architectures*

5. Viara Popova (EUR) *Knowledge discovery and monotonicity*

6. Bart-Jan Hommes (TUD) *The Evaluation of Business Process Modeling Techniques*

7. Elise Boltjes (UM) *Voorbeeldig onderwijs; voorbeeldgestuurd onderwijs, een opstap naar abstract denken, vooral voor meisjes*

8. Joop Verbeek(UM) *Politie en de Nieuwe Internationale Informatiemarkt, Grensregionale politiële gegevensuitwisseling en digitale expertise*
9. Martin Caminada (VU) *For the Sake of the Argument; explorations into argument-based reasoning*
10. Suzanne Kabel (UvA) *Knowledge-rich indexing of learning-objects*
11. Michel Klein (VU) *Change Management for Distributed Ontologies*
12. The Duy Bui (UT) *Creating emotions and facial expressions for embodied agents*
13. Wojciech Jamroga (UT) *Using Multiple Models of Reality: On Agents who Know how to Play*
14. Paul Harrenstein (UU) *Logic in Conflict. Logical Explorations in Strategic Equilibrium*
15. Arno Knobbe (UU) *Multi-Relational Data Mining*
16. Federico Divina (VU) *Hybrid Genetic Relational Search for Inductive Learning*
17. Mark Winands (UM) *Informed Search in Complex Games*
18. Vania Bessa Machado (UvA) *Supporting the Construction of Qualitative Knowledge Models*
19. Thijs Westerveld (UT) *Using generative probabilistic models for multimedia retrieval*
20. Madelon Evers (Nyenrode) *Learning from Design: facilitating multidisciplinary design teams*

2005

1. Floor Verdenius (UvA) *Methodological Aspects of Designing Induction-Based Applications*
2. Erik van der Werf (UM) *AI techniques for the game of Go*
3. Franc Grootjen (RUN) *A Pragmatic Approach to the Conceptualisation of Language*
4. Nirvana Meratnia (UT) *Towards Database Support for Moving Object data*
5. Gabriel Infante-Lopez (UvA) *Two-Level Probabilistic Grammars for Natural Language Parsing*

6. Pieter Spronck (UM) *Adaptive Game AI*
7. Flavius Frasincar (TU/e) *Hypermedia Presentation Generation for Semantic Web Information Systems*
8. Richard Vdovjak (TU/e) *A Model-driven Approach for Building Distributed Ontology-based Web Applications*
9. Jeen Broekstra (VU) *Storage, Querying and Inferencing for Semantic Web Languages*
10. Anders Bouwer (UvA) *Explaining Behaviour: Using Qualitative Simulation in Interactive Learning Environments*
11. Elth Ogston (VU) *Agent Based Matchmaking and Clustering - A Decentralized Approach to Search*
12. Csaba Boer (EUR) *Distributed Simulation in Industry*
13. Fred Hamburg (UL) *Een Computermodel voor het Ondersteunen van Euthanasiebeslissingen*
14. Borys Omelayenko (VU) *Web-Service configuration on the Semantic Web; Exploring how semantics meets pragmatics*
15. Tibor Bosse (VU) *Analysis of the Dynamics of Cognitive Processes*
16. Joris Graaumans (UU) *Usability of XML Query Languages*
17. Boris Shishkov (TUD) *Software Specification Based on Re-usable Business Components*
18. Danielle Sent (UU) *Test-selection strategies for probabilistic networks*
19. Michel van Dartel (UM) *Situated Representation*
20. Cristina Coteanu (UL) *Cyber Consumer Law, State of the Art and Perspectives*
21. Wijnand Derks (UT) *Improving Concurrency and Recovery in Database Systems by Exploiting Application Semantics*

2006

1. Samuil Angelov (TU/e) *Foundations of B2B Electronic Contracting*
2. Cristina Chisalita (VU) *Contextual issues in the design and use of information technology in organizations*

3. Noor Christoph (UvA) *The role of metacognitive skills in learning to solve problems*
4. Marta Sabou (VU) *Building Web Service Ontologies*
5. Cees Pierik (UU) *Validation Techniques for Object-Oriented Proof Outlines*
6. Ziv Baida (VU) *Software-aided Service Bundling - Intelligent Methods & Tools for Graphical Service Modeling*
7. Marko Smiljanic (UT) *XML schema matching – balancing efficiency and effectiveness by means of clustering*
8. Eelco Herder (UT) *Forward, Back and Home Again - Analyzing User Behavior on the Web*
9. Mohamed Wahdan (UM) *Automatic Formulation of the Auditor's Opinion*
10. Ronny Siebes (VU) *Semantic Routing in Peer-to-Peer Systems*
11. Joeri van Ruth (UT) *Flattening Queries over Nested Data Types*
12. Bert Bongers (VU) *Interactivation - Towards an e-cology of people, our technological environment, and the arts*
13. Henk-Jan Lebbink (UU) *Dialogue and Decision Games for Information Exchanging Agents*
14. Johan Hoorn (VU) *Software Requirements: Update, Upgrade, Redesign - towards a Theory of Requirements Change*
15. Rainer Malik (UU) *CONAN: Text Mining in the Biomedical Domain*
16. Carsten Riggelsen (UU) *Approximation Methods for Efficient Learning of Bayesian Networks*
17. Stacey Nagata (UU) *User Assistance for Multitasking with Interruptions on a Mobile Device*
18. Valentin Zhizhkun (UvA) *Graph transformation for Natural Language Processing*
19. Birna van Riemsdijk (UU) *Cognitive Agent Programming: A Semantic Approach*
20. Marina Velikova (UvT) *Monotone models for prediction in data mining*
21. Bas van Gils (RUN) *Aptness on the Web*

22. Paul de Vrieze (RUN) *Fundaments of Adaptive Personalisation*

23. Ion Juvina (UU) *Development of Cognitive Model for Navigating on the Web*

24. Laura Hollink (VU) *Semantic Annotation for Retrieval of Visual Resources*

25. Madalina Drugan (UU) *Conditional log-likelihood MDL and Evolutionary MCMC*

26. Vojkan Mihajlović (UT) *Score Region Algebra: A Flexible Framework for Structured Information Retrieval*

27. Stefano Bocconi (CWI) *Vox Populi: generating video documentaries from semantically annotated media repositories*

28. Börkur Sigurbjörnsson (UvA) *Focused Information Access using XML Element Retrieval*

2007

1. Kees Leune (UvT) *Access Control and Service-Oriented Architectures*

2. Wouter Teepe (RUG) *Reconciling Information Exchange and Confidentiality: A Formal Approach*

3. Peter Mika (VU) *Social Networks and the Semantic Web*

4. Jurriaan van Diggelen (UU) *Achieving Semantic Interoperability in Multi-agent Systems: a dialogue-based approach*

5. Bart Schermer (UL) *Software Agents, Surveillance, and the Right to Privacy: a Legislative Framework for Agent-enabled Surveillance*

6. Gilad Mishne (UvA) *Applied Text Analytics for Blogs*

7. Nataša Jovanović (UT) *To Whom It May Concern - Addressee Identification in Face-to-Face Meetings*

8. Mark Hoogendoorn (VU) *Modeling of Change in Multi-Agent Organizations*

9. David Mobach (VU) *Agent-Based Mediated Service Negotiation*

10. Huib Aldewereld (UU) *Autonomy vs. Conformity: an Institutional Perspective on Norms and Protocols*

11. Natalia Stash (TU/e) *Incorporating Cognitive/Learning Styles in a General-Purpose Adaptive Hypermedia System*
12. Marcel van Gerven (RUN) *Bayesian Networks for Clinical Decision Support: A Rational Approach to Dynamic Decision-Making under Uncertainty*
13. Rutger Rienks (UT) *Meetings in Smart Environments; Implications of Progressing Technology*
14. Niek Bergboer (UM) *Context-Based Image Analysis*
15. Joyca Lacroix (UM) *NIM: a Situated Computational Memory Model*
16. Davide Grossi (UU) *Designing Invisible Handcuffs. Formal investigations in Institutions and Organizations for Multi-agent Systems*
17. Theodore Charitos (UU) *Reasoning with Dynamic Networks in Practice*
18. Bart Orriëns (UvT) *On the development an management of adaptive business collaborations*
19. David Levy (UM) *Intimate relationships with artificial partners*
20. Slinger Jansen (UU) *Customer Configuration Updating in a Software Supply Network*
21. Karianne Vermaas (UU) *Fast diffusion and broadening use: A research on residential adoption and usage of broadband internet in the Netherlands between 2001 and 2005*
22. Zlatko Zlatev (UT) *Goal-oriented design of value and process models from patterns*
23. Peter Barna (TU/e) *Specification of Application Logic in Web Information Systems*
24. Georgina Ramírez Camps (CWI) *Structural Features in XML Retrieval*
25. Joost Schalken (VU) *Empirical Investigations in Software Process Improvement*

2008

1. Katalin Boer-Sorbán (EUR) *Agent-Based Simulation of Financial Markets: A modular, continuous-time approach*
2. Alexei Sharpanskykh (VU) *On Computer-Aided Methods for Modeling and Analysis of Organizations*

3. Vera Hollink (UvA) *Optimizing hierarchical menus: a usage-based approach*
4. Ander de Keijzer (UT) *Management of Uncertain Data - towards unattended integration*
5. Bela Mutschler (UT) *Modeling and simulating causal dependencies on process-aware information systems from a cost perspective*
6. Arjen Hommersom (RUN) *On the Application of Formal Methods to Clinical Guidelines, an Artificial Intelligence Perspective*
7. Peter van Rosmalen (OU) *Supporting the tutor in the design and support of adaptive e-learning*
8. Janneke Bolt (UU) *Bayesian Networks: Aspects of Approximate Inference*
9. Christof van Nimwegen (UU) *The paradox of the guided user: assistance can be counter-effective*
10. Wauter Bosma (UT) *Discourse oriented summarization*
11. Vera Kartseva (VU) *Designing Controls for Network Organizations: A Value-Based Approach*
12. József Farkas (RUN) *A Semiotically Oriented Cognitive Model of Knowledge Representation*
13. Caterina Carraciolo (UvA) *Topic Driven Access to Scientific Handbooks*
14. Arthur van Bunningen (UT) *Context-Aware Querying; Better Answers with Less Effort*
15. Martijn van Otterlo (UT) *The Logic of Adaptive Behavior: Knowledge Representation and Algorithms for the Markov Decision Process Framework in First-Order Domains.*
16. Henriëtte van Vugt (VU) *Embodied agents from a user's perspective*
17. Martin Op 't Land (TUD) *Applying Architecture and Ontology to the Splitting and Allying of Enterprises*
18. Guido de Croon (UM) *Adaptive Active Vision*
19. Henning Rode (UT) *From Document to Entity Retrieval: Improving Precision and Performance of Focused Text Search*
20. Rex Arendsen (UvA) *Geen bericht, goed bericht. Een onderzoek naar de effecten van de introductie van elektronisch berichtenverkeer met de overheid op de administratieve lasten van bedrijven*

21. Krisztian Balog (UvA) *People Search in the Enterprise*
22. Henk Koning (UU) *Communication of IT-Architecture*
23. Stefan Visscher (UU) *Bayesian network models for the management of ventilator-associated pneumonia*
24. Zharko Aleksovski (VU) *Using background knowledge in ontology matching*
25. Geert Jonker (UU) *Efficient and Equitable Exchange in Air Traffic Management Plan Repair using Spender-signed Currency*
26. Marijn Huijbregts (UT) *Segmentation, Diarization and Speech Transcription: Surprise Data Unraveled*
27. Hubert Vogten (OU) *Design and Implementation Strategies for IMS Learning Design*
28. Ildiko Flesch (RUN) *On the Use of Independence Relations in Bayesian Networks*
29. Dennis Reidsma (UT) *Annotations and Subjective Machines - Of Annotators, Embodied Agents, Users, and Other Humans*
30. Wouter van Atteveldt (VU) *Semantic Network Analysis: Techniques for Extracting, Representing and Querying Media Content*
31. Loes Braun (UM) *Pro-Active Medical Information Retrieval*
32. Trung H. Bui (UT) *Toward Affective Dialogue Management using Partially Observable Markov Decision Processes*
33. Frank Terpstra (UvA) *Scientific Workflow Design; theoretical and practical issues*
34. Jeroen de Knijf (UU) *Studies in Frequent Tree Mining*
35. Ben Torben Nielsen (UvT) *Dendritic morphologies: function shapes structure*

2009

1. Rasa Jurgelenaite (RUN) *Symmetric Causal Independence Models*
2. Willem Robert van Hage (VU) *Evaluating Ontology-Alignment Techniques*
3. Hans Stol (UvT) *A Framework for Evidence-based Policy Making Using IT*

4. Josephine Nabukenya (RUN) *Improving the Quality of Organisational Policy Making using Collaboration Engineering*
5. Sietse Overbeek (RUN) *Bridging Supply and Demand for Knowledge Intensive Tasks - Based on Knowledge, Cognition, and Quality*
6. Muhammad Subianto (UU) *Understanding Classification*
7. Ronald Poppe (UT) *Discriminative Vision-Based Recovery and Recognition of Human Motion*
8. Volker Nannen (VU) *Evolutionary Agent-Based Policy Analysis in Dynamic Environments*
9. Benjamin Kanagwa (RUN) *Design, Discovery and Construction of Service-oriented Systems*
10. Jan Wielemaker (UvA) *Logic programming for knowledge-intensive interactive applications*
11. Alexander Boer (UvA) *Legal Theory, Sources of Law & the Semantic Web*
12. Peter Massuthe (TU/e, Humboldt-Universitaet zu Berlin) *perating Guidelines for Services*
13. Steven de Jong (UM) *Fairness in Multi-Agent Systems*
14. Maksym Korotkiy (VU) *From ontology-enabled services to service-enabled ontologies (making ontologies work in e-science with ONTO-SOA)*
15. Rinke Hoekstra (UvA) *Ontology Representation - Design Patterns and Ontologies that Make Sense*
16. Fritz Reul (UvT) *New Architectures in Computer Chess*
17. Laurens van der Maaten (UvT) *Feature Extraction from Visual Data*
18. Fabian Groffen (CWI) *Armada, An Evolving Database System*
19. Valentin Robu (CWI) *Modeling Preferences, Strategic Reasoning and Collaboration in Agent-Mediated Electronic Markets*
20. Bob van der Vecht (UU) *Adjustable Autonomy: Controling Influences on Decision Making*
21. Stijn Vanderlooy (UM) *Ranking and Reliable Classification*
22. Pavel Serdyukov (UT) *Search For Expertise: Going beyond direct evidence*
23. Peter Hofgesang (VU) *Modelling Web Usage in a Changing Environment*

24. Annerieke Heuvelink (VUA) *Cognitive Models for Training Simulations*
25. Alex van Ballegooij (CWI) *RAM: Array Database Management through Relational Mapping*
26. Fernando Koch (UU) *An Agent-Based Model for the Development of Intelligent Mobile Services*
27. Christian Glahn (OU) *Contextual Support of social Engagement and Reflection on the Web*
28. Sander Evers (UT) *Sensor Data Management with Probabilistic Models*
29. Stanislav Pokraev (UT) *Model-Driven Semantic Integration of Service-Oriented Applications*
30. Marcin Zukowski (CWI) *Balancing vectorized query execution with bandwidth-optimized storage*
31. Sofiya Katrenko (UvA) *A Closer Look at Learning Relations from Text*
32. Rik Farenhorst (VU) and Remco de Boer (VU) *Architectural Knowledge Management: Supporting Architects and Auditors*
33. Khiet Truong (UT) *How Does Real Affect Affect Affect Recognition In Speech?*
34. Inge van de Weerd (UU) *Advancing in Software Product Management: An Incremental Method Engineering Approach*
35. Wouter Koelewijn (UL) *Privacy en Politiegegevens; Over geautomatiseerde normatieve informatie-uitwisseling*
36. Marco Kalz (OU) *Placement Support for Learners in Learning Networks*
37. Hendrik Drachsler (OU) *Navigation Support for Learners in Informal Learning Networks*
38. Riina Vuorikari (OU) *Tags and self-organisation: a metadata ecology for learning resources in a multilingual context*
39. Christian Stahl (TU/e, Humboldt-Universitaet zu Berlin) *Service Substitution – A Behavioral Approach Based on Petri Nets*
40. Stephan Raaijmakers (UvT) *Multinomial Language Learning, Investigations into the Geometry of Language*

TiCC Ph.D. Series

1. Pashiera Barkhuysen
   Audiovisual prosody in interaction
   Promotores: M.G.J. Swerts, E.J. Krahmer
   Tilburg, 3 October 2008

2. Ben Torben-Nielsen
   Dendritic morphology: function shapes structure
   Promotores: H.J. van den Herik, E.O. Postma
   Co-promotor: K.P. Tuyls
   Tilburg, 3 December 2008

3. Hans Stol
   A framework for evidence-based policy making using IT
   Promotor: H.J. van den Herik
   Tilburg, 21 January 2009

4. Jeroen Geertzen
   Act recognition and prediction. Explorations in computational dialogue modelling
   Promotor: H.C. Bunt
   Co-promotor: J.M.B. Terken
   Tilburg, 11 February 2009

5. Sander Canisius
   Structural prediction for natural language processing: a constraint satisfaction approach
   Promotores: A.P.J. van den Bosch, W.M.P. Daelemans
   Tilburg, 13 February 2009

6. Fritz Reul
   New Architectures in Computer Chess Dutch information only
   Promotor: H.J. van den Herik

Co-promotor: J. Uiterwijk
Tilburg, 17 June 2009

7. Laurens van der Maaten
Feature extraction from visual data
Promotores: E.O. Postma, H.J. van den Herik
Co-promotor: A.G. Lange
Tilburg, 23 June 2009

8. Stephan Raaijmakers
Multinomial language learning, Investigations into the geometry of language
Promotores: A.P.J. van den Bosch, W.M.P. Daelemans
Tilburg, 1 December 2009